A FORMAL FRAMEWORK FOR OPTIMIZING AND EVALUATING INTERACTIVE RETRIEVAL SYSTEM

BY

YINAN ZHANG

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science in the Graduate College of the University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Doctoral Committee:

Professor ChengXiang Zhai, Chair
Professor Jiawei Han
Professor Aditya Parameswaran
Professor Norbert Fuhr, University of Duisburg-Essen
ABSTRACT

The past decades have seen dramatic increase in the amount of information available to us, and the research area of information retrieval has served to help us access the tiny subset of information relevant to each of us more efficiently and more effectively. This thesis studies how to formally optimize as well as evaluate an interactive information retrieval system.

First, we propose a formal general framework, the Interface Card Model, for optimizing interactive retrieval interface. We frame the task of an interactive retrieval system as to choose a sequence of interface cards to present to the user that can maximize the expected gain of relevant information for the user while minimizing the effort of the user, with consideration of the user’s action model and any desired constraints on the interface card. We show that such a formal Interface Card Model can not only cover the classic Probability Ranking Principle as a special case by making multiple simplification assumptions, but also be used to derive a novel formal interface model for adaptively optimizing navigational interfaces in a retrieval system.

Second, we propose a novel formulation of the Interface Card Model, the Interface Card Model with User States, for solving concrete interface optimization problems. The formulation is based on sequential decision theory, leading to a general framework for formal modeling of user states and stopping actions. Simulation and user study experiments demonstrate the effectiveness of the proposed model in automatically adjusting the interface layout in adaptation to inferred user stopping tendencies in addition to user interaction and screen size.

Third, as a specific example of applying our proposed interface optimization framework in a larger scale real world application, we propose a Bayesian framework for user preference modeling and dynamically optimizing a faceted browsing system based on users’ facet selection interactions.

Finally, we propose a general formal framework for evaluating IR systems
based on search session simulation that can be used to perform reproducible experiments for evaluating any IR system, including interactive systems and systems with sophisticated interfaces. We show that the traditional Cranfield evaluation method can be regarded as a special instantiation of the proposed framework where the simulated search session is a user sequentially browsing the presented search results. We further show that the proposed framework enables us to evaluate a set of tag-based search interfaces, a generalization of faceted browsing interfaces, producing results consistent with real user experiments and revealing interesting findings about effectiveness of the interfaces for different types of users.
ACKNOWLEDGMENTS

I want to express my sincere gratitude to my advisor Professor ChengXiang Zhai for his tremendous support and guidance through my PhD study. He is a visionary researcher who always guides me with unique insights into research problems, and he is a very kind and patient mentor who never prioritizes speed over my interest. Above all, his great passion for research from the bottom of his heart has a strong influence on me, in the past as well as many more years to come.

I would also like to thank my committee members, Professor Jiawei Han, Professor Aditya Parameswaran, and Professor Norbert Fuhr, for their invaluable feedback on my research work. Professor Han gave me a lot of constructive comments from the perspective of data mining and user modeling in the real world setting. Professor Parameswaran provided me many great suggestions for expanding the thesis work to connect to research works on a broader range of information systems. Professor Fuhr helped me establish a deep and systematic view of the contribution and limitation of the thesis work and its theoretical relationship to former formal frameworks in information retrieval research.

I am very grateful to Professor Saurabh Sinha, my initial advisor. He kindly supported me during the initial year of my study, and I was greatly inspired by his knowledge and enthusiasm in biological and bioinformatic research. Many thanks to Jim Hahn, who provided me gracious support during the early time of my study. I also want to thank the very kind staff in the computer science department, especially (alphabetically) Donna Coleman, Mary Beth Kelley and Kara Lynn MacGregor, for their kind help in a lot of aspects through my PhD years.

I owe gratitude to my friends in UIUC. It has been very enjoyable to work and have fun in the TIMAN group with (alphabetically) Huizhong Duan, Chase Geigle, Shan Jiang, Yanen Li, Xueqing Liu, Ismini Lourentzou, Sean
Massung, Mingjie Qian, Parikshit Sondhi, Hongning Wang, Sheng Wang, and Rongda Zhu. I also want to thank my other friends at UIUC, especially (alphabetically): Charles Blatti, Thyago Duque, Huan Gui, Casey Hanson, Ming Ji, Majid Kazemian, Jialu Liu, Nathan Ryckman, Abul Hassan Samee, Yizhou Sun, Kun Tian, Chi Wang, Carl Yang, Chao Zhang, and Mianwei Zhou. I hope the graduation does not separate us from each other.

Finally and above all, I am deeply indebted to my parents and my wife Jing. I want to thank my parents for their unconditional love to me in every aspect in my life, which gave me strength and courage to stand firm and embrace challenges. I want to thank my wife Jing for the crucial love, caring and support throughout my PhD study; it is very hard to imagine how these years would have been possible without her.
To the human race.
# TABLE OF CONTENTS

**CHAPTER 1** INTRODUCTION ........................................... 1  
**CHAPTER 2** RELATED WORK ................................. 6  
**CHAPTER 3** INTERFACE CARD MODEL ......................... 11  
  3.1 Interface Card Model ........................................... 12  
  3.2 Plain Card .................................................. 17  
  3.3 Navigational Card ........................................... 20  
  3.4 Summary .................................................. 34  
**CHAPTER 4** SEQUENTIAL DECISION FORMULATION OF INTERFACE CARD MODEL ........................................ 36  
  4.1 Interface Card Model with User States ..................... 37  
  4.2 Stopping Action Modeling ................................... 44  
  4.3 Plain Card .................................................. 47  
  4.4 Navigational Card ........................................... 51  
  4.5 Summary .................................................. 61  
**CHAPTER 5** BAYESIAN FRAMEWORK FOR USER PREFERENCE IN INTERACTIVE INFORMATION RETRIEVAL ............................. 63  
  5.1 A Bayesian Probabilistic Framework ....................... 63  
  5.2 Action Model and Inference ................................ 65  
  5.3 Experiments ............................................... 71  
  5.4 Summary .................................................. 72  
**CHAPTER 6** SEARCH SIMULATION FRAMEWORK FOR EVALUATING INTERACTIVE INFORMATION RETRIEVAL .............................. 74  
  6.1 Search Simulation Framework ................................. 76  
  6.2 Analysis of Existing Metrics ............................... 80  
  6.3 Simulated Evaluation on Tag-based Search Interface ........ 84  
  6.4 Summary .................................................. 93  
**CHAPTER 7** CONCLUSION ............................................. 94  
**REFERENCES** .................................................. 96
The past decades have seen dramatic increase in the amount of information available to us, and the research area of information retrieval (IR) has served to help us access the tiny subset of information relevant to each of us more efficiently and more effectively. Developing formal models for information retrieval (IR) has always been an important fundamental challenge. For example, the Probability Ranking Principle (PRP) [1] proposed more than three decades ago has laid out a solid foundation and provided a theoretical justification for framing the retrieval task as a ranking problem, leading to the development of many effective retrieval functions for ranking documents that are used in current search engines (e.g., [2, 3, 4, 5, 6, 7, 8]). Despite the great success of PRP, however, it is also known that it is based on two problematic assumptions, i.e., sequential browsing and independent relevance (utility) of documents, which are generally not true in practice. As a result, e.g., the traditional retrieval models developed based on PRP cannot handle redundancy among documents directly (and must rely on post-processing of search results), an immediate consequence of the independence assumption of document relevance. The sequential browsing assumption implies limitations on the interface also, and in particular ignores the actions that a user can take when interacting with an interface displaying search results (e.g., faceted browsing).

Recognizing these limitations and attempting to generalize the PRP for interactive IR, Fuhr[9] has recently proposed a novel formal framework for optimizing interactive retrieval and derived a PRP for interactive IR (IIR-PRP) where a user’s effort and benefit are captured when optimizing the ranking of documents. This work effectively addressed the independence assumption made in the PRP and provides a theoretical foundation for optimizing document ranking when a user is assumed to interactively browse
a list of search results. Meanwhile, it assumes that the interaction between the user and the system could be decomposed into a sequential list of binary choices in a clear and unambiguous manner, a more generalized version of the sequential browsing assumption, which remains an assumption made for optimizing ranking in interactive retrieval. In this thesis, we relax this assumption and propose a more general formal model than IIR-PRP for optimizing interactive retrieval.

The sequential browsing assumption touches a much larger problem of how to model a user’s reactions to a retrieval result interface, which further depends on what the interface looks like, raising the interesting question “how can we formally model the problem of interface design for an interactive IR system?” Interestingly, in contrast with a large body of work on formal methods for optimizing ranking, there has been little work on formal methods for optimizing the interface of a system, despite that the dynamic and interactive nature of information seeking process has long been recognized and studied from information science perspective (see, e.g., [10, 11]). While optimizing ranking is clearly very important for optimizing a retrieval system, when we consider optimizing an interactive retrieval system, we must also optimize the interface part of the system so as to optimize the whole system, which is the goal of our study.

The study of interface optimization is especially important in the current era of ever faster technology advancement, leading to the emergence of smartphones and various kinds of wearable devices with very small screens, which generally require a different interface than the popular interface designed for desktops. For example, while showing a document list on a relatively large screen is popular and appropriate, it may not be appropriate to do so on a very small screen where an interface with navigational tags might be more useful as it enables a user to navigate into the relevant information more efficiently. Even if we consider the current interface of a Web search engine such as Google or Bing on a large screen, which typically shows a list of fixed number of snippets on each page, there are still many interesting questions related to the optimization of the interface. For example, how many snippets should we display on each page? The commonly used number, 10, is not necessarily always the best choice. Also, what about shortening some
snippets to make room for more results or vice versa? These questions have been tackled by Human-Computer Interaction (HCI) researchers with many empirical findings. Unfortunately, it is still unclear how we can leverage these findings to build an intelligent IR system that can automatically optimize its interaction interface adaptively both to the screen size and to the user’s information need.

In this thesis, we systematically study how to optimize as well as evaluate an interactive retrieval system, and we make the following contributions.

First, we propose a formal general framework, the Interface Card Model, for optimizing interactive retrieval interface [12]. We frame the task of an interactive retrieval system as to choose a sequence of interface cards to present to the user. At each interaction lap, the system’s goal is to choose an interface card that can maximize the expected gain of relevant information for the user while minimizing the effort of the user with consideration of the user’s action model and any desired constraints on the interface card. We show that such a formal Interface Card Model can not only cover the Probability Ranking Principle for Interactive Information Retrieval as a special case by making multiple simplification assumptions, but also be used to derive a novel formal interface model for adaptively optimizing navigational interfaces in a retrieval system. Experimental results show that the proposed model is effective in automatically generating adaptive navigational interfaces, which outperform the baseline pre-designed static interfaces. We elaborate more about the Interface Card Model in Chapter 3.

Second, we propose a novel formulation of the Interface Card Model (ICM), the Interface Card Model with User States (ICM-US), as a formal instantiation of ICM for solving concrete interface optimization problems [13]. The ICM-US model is based on sequential decision theory, leading to a general framework for formal modeling of user states and stopping actions. The proposed formulation naturally connects optimization of interactive retrieval with Markov Decision Processes and Partially Observable Markov Decision Processes, and enables the use of reinforcement learning algorithms for optimizing interactive retrieval interfaces. Simulation and user study experiments demonstrate the effectiveness of the proposed model in automatically adjusting the interface layout in adaptation to inferred user stopping tendencies in
addition to user interaction and screen size. We elaborate more about the ICM-US model in Chapter 4.

Third, as a specific example of applying our proposed interface optimization framework in a larger scale real world application, we propose a Bayesian framework for user preference modeling and dynamically optimizing a faceted browsing system based on users’ facet selection interactions [14]. Simulation experiment results with product search log show promise of the framework, which opens up interesting opportunities for new research in the intersection of machine learning, information retrieval and economics. We elaborate more about the Bayesian framework for interactive retrieval optimization in Chapter 5.

Finally, we propose a formal evaluation framework for systematically evaluating an interactive retrieval interface in general [15]. In traditional information retrieval evaluation, human judged test collections such as the Cranfield test collection is used as a gold standard for assessing the effectiveness of retrieval systems, but they are restricted to evaluating ranking systems and could not be easily extended to evaluate interactive retrieval systems in general. On the other hand, user studies are widely employed for comparing the effectiveness of different systems or different retrieval strategies in more sophisticated settings, but they suffer from the issue of high cost and low repeatability. Many simulation evaluation schemes were proposed in the past that incur very low cost and are highly repeatable, such as the ones used in [14] and [13], which could also be used for evaluating relatively more sophisticated interactive retrieval systems, but they are each designed for a specific retrieval system in an ad hoc manner. We propose a general formal framework for evaluating IR systems based on search session simulation that can be used to perform reproducible experiments for evaluating any IR system, including interactive systems and systems with sophisticated interfaces. We show that the traditional Cranfield evaluation method can be regarded as a special instantiation of the proposed framework where the simulated search session is a user sequentially browsing the presented search results. By examining a number of existing evaluation metrics in the proposed framework, we reveal the exact assumptions they have made implicitly about the simulated users and discuss possible ways to improve these metrics. We further show
that the proposed framework enables us to evaluate a set of tag-based search interfaces, a generalization of faceted browsing interfaces, producing results consistent with real user experiments and revealing interesting findings about effectiveness of the interfaces for different types of users. We elaborate more about our proposed evaluation framework in Chapter 6.

We conclude the thesis and discuss about its impact and the roadmap for future research in Chapter 7.
The majority work on formal models for Information Retrieval (IR) has been based on the Probability Ranking Principle (PRP) [1] and all attempt to optimize a ranking function defined on a query-document pair; they include all kinds of traditional retrieval models such as vector space models [2], classic probabilistic models [3], language models [4, 5, 16], divergence from randomness [6], inference networks [7], axiomatic approaches [8], and recent extensions in the direction of learning to rank [17, 18]. These works generally do not model user interactions, thus providing no guidance for interface design.

The PRP for interactive IR (IIR-PRP) [9] generalizes the PRP to optimize ranking in an interactive IR setting, where a number of important concepts for modeling user interaction from the perspective of decision making were introduced, including situation, effort and benefit, and an optimal ordering principle is derived for ranking items when a user is assumed to sequentially make “accept/reject” decisions through the list. Our model shares a similar high-level goal with the IIR-PRP in that both attempt to establish a formal model for interactive retrieval, but it is more general than the IIR-PRP, which can be shown as a special case of our model under a set of simplification assumptions. Due to its generality, our model can be directly used to optimize the interaction interface; as a result, our model can suggest interfaces that would dynamically adapt to the assumed screen sizes, which cannot be achieved in any existing work on formal models.

The dynamic and interactive nature of information seeking process has long been recognized and studied from information science perspective (see, e.g., [10, 11]). Our work can be regarded as an attempt to formalize some of the theoretical arguments in these works with an operational mathematical model that can be used for building an intelligent IR system with an adaptive
interface for navigation.

Our model of dynamic information need is related to the ostensive model (OM) [19], which provides a framework for modeling the evolution of information needs over time. Our model is sufficiently general to allow us to refine it with the ostensive model or any other model of evolving information needs. Our proposed framework enables any such model to be adopted for optimizing navigational interface.

The model we derived for optimizing navigational interface uses an objective function to maximize the difference between a user’s benefit and cost for finding a relevant information item. Such a decision criterion is related to some recent works that have explored economic models for IR (e.g., [20, 21]). Furthermore, optimizing the ranking of documents with consideration of user actions has also been studied in the context of feedback to optimize the session-level utility [22] and using a POMDP framework in [23], where a dual-agent POMDP was proposed to model both user actions and system actions. However, none of these studies has proposed a model to optimize navigational interface, a primary goal of this thesis.

Optimizing search engine interface has been extensively studied in the Human-Computer Interaction community (see, e.g., the survey in the book [24]), including designing and evaluating faceted browsing systems and coming up with various ad hoc ways to optimize such systems (e.g., [25] and [26]). However, no existing work can optimize a navigational interface with an explicitly defined objective function, which our model attempts to achieve.

A popular and interesting approach frequently used in contemporary information access interfaces is the tag cloud, where a two-dimensional “cloud” of representative phrases associated with the document collection is displayed, and the users could click one of the shown phrases to zoom into the document subsets associated with the phrase [27, 28, 29, 30]. Such novel interfaces are found in research studies to be especially useful when the users’ information-seeking tasks are relatively general [31]. Our proposed Interface Card Model serves as a more fundamental theoretical framework for optimizing interactive retrieval interfaces, and could be applied to optimize creations of tag clouds by properly defining and estimating the reward/cost and the user
action model for clicking on the tags.

The sequential decision formulation of the Interface Card Model is related to several lines of recent work on POMDP and reinforcement learning (e.g., [23, 32, 33]) and economic IR models (e.g., [20, 34]), especially because all these works tend to also model user interactions formally. The main difference between our formulation and these work is that our formulation is based on the Interface Card Model, which is a very general framework for modeling retrieval process (framed as choosing optimally a sequence of interface cards), thus our formulation can potentially model and optimize very complicated interaction interfaces, whereas the other works cannot optimize the design of an interface due to the restriction to mostly a ranking-based formulation of the retrieval problem. However, the specific techniques and models proposed in these existing work can all potentially contribute to further refinement of our formulation to make it even more operational. For example, the economic IR work would enable our formulation to incorporate a user decision model for refining the user actions (e.g., modeling how the user actions depend on the user state).

Our user action model characterizes how users make navigational decisions and stopping decisions in an information seeking process, which is related to Information Foraging theories [35], where models of user actions driven by information scent were proposed to describe how users navigate on the web following hyperlinks. Recent works in user search behavior analysis proposed and evaluated novel models for users’ navigation actions [36], stopping actions [37], and for characterizing users’ patience levels [38] in a search session. Our framework is more focused on the “orthogonal” question of how to optimize interactive search interfaces given a learned user action model; such line of research and the existing works in user action modeling serve to complement each other and collectively lead to more effective search systems for users.

Evaluation has always been a central research topic in IR; the three surveys by Sanderson [39], Kelly [40], and Harman [41] have covered most progress in IR evaluation research, though many newer papers on the topic have been published since those three surveys were written, notably the axiomatic approaches to IR evaluation [42], and applications of statistical analysis techniques. Cranfield test-collection evaluation methodology proposed a long
time ago [43] remains the dominant evaluation method in IR for comparing
different retrieval algorithms today, and the ranking performance is often
assessed using measures such as Precision, Recall, MAP and/or NDCG. Ad-
ditional evaluation measures have been proposed and used for evaluating
various IR tasks, such as $\alpha$-NDCG [44], Rank-based Precision [38], Expected
Reciprocal Rank [45], and time-based measure [46]. However, while these ap-
proaches work well for evaluating retrieval results in the form of a ranked list,
it is unclear how it can be applied to evaluate an interactive retrieval system.
The proposed simulation-based evaluation framework breaks this limitation
and generalizes the current Cranfield evaluation method to provide a prin-
cipled way to evaluate any interactive system. It was demonstrated in [47]
that MAP could be derived under certain user behavior assumptions. We
show that the proposed evaluation framework is a more general framework
on user behavior simulation that can cover all aforementioned measures as
well as many other measures as special cases.

User studies are also often conducted to evaluate an IR system, including
both small-scale controlled studies and larger-scale user studies using A/B
test. While such an evaluation method involves real users and accurately
reflects the utility of a retrieval system in application settings, it has a seri-
ous drawback (as compared with Cranfield evaluation method) in not being
reproducible. A main point of our proposed evaluation framework is that the
only way to enable reproducible experiments with interactive IR systems is
through user simulation. The framework can be regarded as both a general-
ization of the test collection approach to enable evaluation of interactive IR
systems, and an “artificial” way to perform interactive user studies.

A previous work [48] has already made an attempt to evaluate session
search by doing simulation; our work is a step forward to propose a more
general framework. Indeed, it appears that we have no choice but to use
such a simulation framework if we want to perform reproducible experiments
to evaluate an interactive retrieval system with sophisticated interfaces since
this appears to be the only way to control the user. Our work is also related
to the recent work by the Glasgow group on user simulation (see, e.g., the
simulation toolkit [49]), but our goal of doing simulation is different, i.e., it
is to evaluate an arbitrary IR system.
There have been extensive studies on evaluating ranking systems’ performance using simulated user [50, 51, 48]. Traditional IR studies have long been focusing on modeling users’ click behaviors [52] and relevance feedback [51, 53]. Recent studies have gone beyond click models to simulate other aspects of user behavior, including simulating user queries [50] (often based on language models [50, 54, 55]), simulating a user’s stopping behavior [56, 57] based on gain/cost ratio [35], and query reformulation [48]. A common weakness of these studies is that they are mostly based on random sampling instead of learning from real user behavior [48]. As a result, it remains a challenge how to fairly compare different algorithms using results generated by these simulators. However, they can be leveraged to build an accurate simulator for use in the proposed evaluation framework.
CHAPTER 3

INTERFACE CARD MODEL

Formal models for optimizing information retrieval systems have been studied for several decades in information retrieval community. Past studies (e.g., [2, 3, 4, 5, 6, 7, 8]) largely rely on strong assumptions about the interface and the user-system interactions in the retrieval process, which make the computation more tractable yet often unrealistic. In particular, existing formal frameworks for information retrieval have mostly focused on optimizing the ranking of documents with little attention paid to the optimization of the navigational interface of a retrieval system, which is an important component in any interactive IR system.

We propose to study the novel problem of automatic interface optimization formally, and introduce a new general formal model, called the Interface Card Model, for optimizing interactive retrieval interface [12]. The basic idea behind this model is to view an interactive retrieval process as a process of the retrieval system playing a cooperative card game with the user in the following way: at each interaction lap, facing a current retrieval context, the system would choose an optimal “interface card” to present to the user. The user can then perform any action from a set of possible actions associated with the interface card presented. Depending on the user’s action on the interface card (e.g., selecting a particular facet value), the system would then transition to a new context, and have to choose another (generally new) interface card to show to the user. The game would continue in such a way until the user decides to stop (either due to satisfaction of the information need or abandoning the search). At each interaction lap, the system’s goal is to choose an interface card that can maximize the expected gain of relevant information for the user while minimizing the effort of the user with consideration of a user action model and any desired constraints on the interface card.
We show that such a general formal Interface Card Model can not only cover IIR-PRP as a special case by making multiple simplification assumptions, but also be used to derive a novel formal interface model for adaptively optimizing navigational interfaces in a retrieval system by assuming that an interface card is composed of one or more information blocks to support interactive navigation and a user’s action is mainly to select one of the presented blocks. The derived model enables, for the first time, automatic generation of optimal navigational interfaces that can be adaptive to screen sizes and user interactions. Experimental results show that the proposed model is effective in automatically generating adaptive navigational interfaces, which outperform the baseline pre-designed static interfaces.

3.1 Interface Card Model

In general, any interaction between a user and an interactive information retrieval system can be partitioned into a series of interaction laps, in each of which the user issues an action and the system then reacts to the user’s action by selecting an optimized interface instance to show to the user. For example, in a traditional search engine, the first interaction lap consists of the user issuing a query and the search engine responding with 10 most relevant items as the first result page. After this interaction lap, the user may issue a second action by either clicking an item or “next page,” and the interface reacts by displaying a second interface instance optimized for the perceived user action.

The interaction laps may be defined in various levels of granularity and the set of user actions would change accordingly. The previous example can be regarded as modeling the interaction at the page level. If, however, the 10 search results could not simultaneously fit into the screen of the interface as in the case of searching with a smart phone, then the interaction can be modeled at a finer granularity - the current screen shown to the user, and the user actions would additionally include scrolling up/down, to which the interface reacts by “sliding” the screen up/down by one position. In this scenario, when the user scrolls down, the interface in theory could have a
chance to decide again according to the user’s action about the item to be
shown next, which may be different from the one originally ranked at this
position. Such “drilling down” of the interaction granularity could continue,
if we consider a user’s every eye movement as an action and the interface
may dynamically change the displayed content accordingly, assuming the
availability of an eye-tracker device.\footnote{In theory, we could go even deeper: imagine that we might some day have sensors
installed for everyone to track every neuron excitement in their brain and consider that
as an interaction unit.}

How do we model an arbitrary interactive retrieval system formally at any
given interaction granularity level? To address this question, we propose to
view any user-interface interaction as a card game, in which the “interface
player” determines the optimal card to play in each lap, and we present a
novel Interface Card Model to formally model the interactive retrieval task.
Unlike a real card game where players maximize their own benefits, however,
the Interface Card Model assumes a cooperative game in which the “interface
player” always maximizes the user’s benefit by taking into consideration the
user’s current action, the interaction history, the reward and cost of the user’s
next possible actions and any constraints posed onto the card the “interface
player” plays at the current lap. We now formally introduce the model by first
defining all these components and then the core mathematical optimization
problem.

**Definition 3.1** (Lap). A *lap* is the interaction unit between the user and
the interface in which the user issues an action and the interface then reacts
by generating an optimized interface instance: $t = 1, 2, \ldots$

The laps serve as the timestamps for the user-system interactions and will
always be shown as superscripts.

**Definition 3.2** (Card). A *card* is an interface instance generated by the
interface system in reaction to the user action in each lap: $q^t$.

The notion of card generalizes a wide range of interface instances including
a result page or a screen of a partial result page in a search engine, a question
in a conversational retrieval interface the system uses to clarify the user’s information need, etc.

**Definition 3.3 (Constraint).** The *constraint* is a possible set of restrictions a card needs to satisfy in a lap, and for simplicity is assumed to have the form of a single constraint function: \( f_t(q^t) \leq 0 \).

The constraint is typically associated with the design and restrictions of the interface. For instance, a result page of a traditional search engine may display at most 10 items at a time. If screens are considered as a finer unit for the interaction, when the user scrolls down, all the items on the next screen except the bottom one are restricted to be the ones sliding from the previous screen. In more complicated interface designs like faceted browsing interfaces, there might be panels of facet values and items regulating how much space they could respectively occupy.

In many cases, the constraint could not be captured within a single constraint function. The notion of the single constraint function is only meant for the purpose of notational simplicity, and more complicated forms of constraint do not change the model in any fundamental way.

**Definition 3.4 (Action).** An *action* is a move the user chooses to take next from a set of possible moves that may depend on the current card: \( a^{t+1} \in \mathcal{A}(q^t) \).

The interaction is initiated either by the user or the system, and we allow both situations in this model. Most of the time, the user is the initiator, and the interaction starts with \( a^1 \) (followed by the interface playing the first card \( q^1 \), then the user issuing the second action \( a^2 \), etc.). For example, when a user queries a search engine, \( a^1 \) would be the very first query the user enters. Alternatively, if the search engine attempts to display a possibly personalized search homepage to each user and at each time, then we could define \( a^1 \) to be the user’s action of entering the website. In both cases, the first card the interface plays is designated to be the first interface instance that needs to be optimized according to the user’s action. Typically, we are not interested in the set of possible actions for the very first user action \( a^1 \) because \( a^1 \) is
always regarded as given to the model and there isn’t any uncertainty in it.

There are occasionally situations where the interface system is the initiator of the interaction, e.g. if a smart phone is set to alert the user whenever some interesting news events happen by popping up a screen of news event snippets. In such a scenario, we set $q^1$ to be this first screen and set $a^1$ to be a “null” action.

For the sake of simplicity, from now on we always assume that the action set $A(q^t)$ is either finite or countably infinite, so that we could use the summation sign “$\sum$” for summing over all possible actions in a particular lap. In cases where the action set is not countable (e.g. if a touch-screen smart phone measures how much force the user uses when touching the screen), we could replace the “$\sum$” signs with the “$\int$” sign and the core model is not affected in any fundamental way.

**Definition 3.5 (Context).** The context is all the information the interface system has accumulated till a particular lap about the user for estimating the user’s choice on the next card: $c^t$. Such information includes (a) a priori information about the user, $i$, if any, (b) interactions in all previous laps, if any, between the user and the interface (i.e. the sequence of previous actions issued by the user and previous cards played by the interface), and (c) the action the user just issued in the current lap. The context is expressed as a vector starting with $c^1 = (i, a^1)$ and iteratively updated by $c^{t+1} = (c^t, q^t, a^{t+1})$.

Typically, the a priori information about the user may capture any prior belief about the user, e.g. any available personalization information.

**Definition 3.6 (Action Model).** The action model specifies the system’s estimated probability distribution of the user’s actions in the next lap given the current card and under the current context: $p(a^{t+1}|c^t, q^t)$.

Here we are assuming a probabilistic model for user actions, which provides a general solid framework for formally modeling most real world scenarios.
**Definition 3.7** (Reward). The *reward* is the system’s estimated expected benefit to the user for issuing an action given the current card and under the current context: $r(a^{t+1}|c^t, q^t)$.

The reward may capture the short-term benefit to the user from a relevant item, as well as any long-term benefit, e.g., if the action serves to navigate the user to a new information subspace (as in the case of answering a clarification question in a conversational retrieval system or clicking a facet value in a faceted browsing system). The reward may depend on future laps if the system decides to perform the estimation computation in such a way, but here for notational simplicity, we only put $c^t$ and $q^t$ into $r(a^{t+1}|c^t, q^t)$ and hide any possible dependency of the reward on future laps in the reward function $r$.

**Definition 3.8** (Cost). The *cost* is the system’s estimated expected effort the user spends for issuing an action given the current card and under the current context: $s(a^{t+1}|c^t, q^t)$.

For example, the cost function typically captures any possible effort the user needs to take for scanning through a result page of a search engine, for the decision-making process to determine whether to click or skip a particular item, etc.

**Definition 3.9** (Surplus). The *surplus* is the difference between the reward and the cost to the user for issuing an action given the current card and under the current context: $u(a^{t+1}|c^t, q^t) = r(a^{t+1}|c^t, q^t) - s(a^{t+1}|c^t, q^t)$.

Here we borrow the concept of surplus from economics studies to designate the net benefit to the user for issuing an action. From the user’s perspective, they would typically tend to choose actions leading to higher surplus, and there have been well established economics theories, e.g. the discrete choice model [58], for modeling such behaviours. In this study, however, we do not go deeper in such directions and stop at the level of the action model in formalizing the user’s behavior from the interface system’s perspective.
With all the necessary components defined, we formally introduce the core mathematical optimization problem:

**Definition 3.10 (Interface Card Optimization).** In each lap \( t \), the interface system should play a card \( q^t \) that maximizes the expected surplus \( u^t \) given the current context and under the current constraint, where the expectation is taken with respect to the user action model:

\[
\begin{align*}
\text{maximize} & \quad E(u^t|c^t, q^t) \\
& = \sum_{a^{t+1} \in \mathcal{A}(q^t)} p(a^{t+1}|c^t, q^t) u(a^{t+1}|c^t, q^t) \\
\text{subject to} & \quad f^*_C(q^t) \leq 0
\end{align*}
\] (3.1)

A possible source of confusion is that there are in total two levels of expectations in this formalism: the inner one being encapsulated within the notion of surplus that deals with the uncertainty in the reward and cost for individual actions (recall that the reward and cost are both defined as expectations), and the outer one that deals with the uncertainty in the user’s decision on which action to issue. Such an encapsulation generally holds in reality and lays down a convenient formalism framework for multiple ways of instantiating the Interface Card Model as we will further discuss in the following sections.

### 3.2 Plain Card

As our first instantiation of the Interface Card Model, we will show that the Interface Card Model can cover the IIR-PRP model as a special case under a set of simplification assumptions. Consider the problem setting of a generalized interactive information retrieval (IIR) system as introduced in [9], where the system’s task is to present a sequential list of binary choices to the user, and the system needs to determine the optimal order of the list so as to maximize the user’s expected benefit. Such a problem formulation generalizes a wide range of IIR tasks. (Please refer to [9] for more in-depth explanations.)
In order to instantiate the IIR-PRP model, we establish the following pair of assumptions:

**Assumption 3.1** (Plain Card). Each card is defined to be a choice in the ranked list. The choices are sequentially denoted by \( e^t, t = 1, 2, \ldots \), and we define \( q^t = e^t \).

**Assumption 3.2** (Binary Action). There are two possible user actions in each lap: \( \mathcal{A}(e^t) = \{a_{0}^{t+1}, a_{1}^{t+1}\} \), where \( a_{0}^{t+1} \) and \( a_{1}^{t+1} \) respectively represent the actions of accepting \( e^t \) and rejecting \( e^t \) (to examine the next choice \( e^{t+1} \)).

The interface card optimization problem is now equivalent to determining which choice to place on each position of the list.

Since a card is simply assumed to be a choice on the list, there is no interesting constraint defined on the interface. Further, we adopt the independence assumption in [9] and assume that the probability of the user accepting a choice is independent of the choices they have rejected, so that the action model does not depend on any previous cards or user actions until an accept action takes place. We also follow [9] to focus on the optimization problem before the user’s first accept action (the optimization problem afterwards is regarded as a new round of optimization), so the context is simply collapsed to the a priori information about the user (if any). We thus omit the notion of context in the following writing, and define the shorthand notation \( p(e^t) \) for specifying the action model: \( p(e^t) = p(a_{0}^{t+1}|e^t) = 1 - p(a_{1}^{t+1}|e^t) \). (Please refer to [9] for the notion of “situations” to understand the details and rationales of these assumptions.)

**Assumption 3.3** (Rejection reward). The reward for a reject action is the expected surplus in the next lap:

\[
\begin{align*}
  r(a_{1}^{t+1}|e^t) &= E(u^{t+1}|e^{t+1}) \\
  \text{(3.2)}
\end{align*}
\]

Here we choose to explicitly model the dependency of the reward in the current lap on the future laps, so in order to optimize the first card, we would need to simultaneously optimize all the following cards (i.e. all choices in the
We define another shorthand notation \( r(e^t) \) for the reward of accepting choice \( e^t \): \( r(e^t) = r(a_{0}^{t+1}|e^t) \). In [9], this reward is further decomposed into the expectation of two cases - (a) the accept action is right and (b) the accept action is wrong. We do not go further along this direction; the main line of the derivation would not be affected in any fundamental way.

**Assumption 3.4** (Decision cost). The costs for the accept and reject actions are the same in each lap, which equal the amount of effort the user spends to examine the current choice for deciding whether they should accept or reject it:

\[
s(a_{0}^{t+1}|e^t) = s(a_{1}^{t+1}|e^t) = s(e^t)
\]  

(3.3)

Now, with all the necessary assumptions introduced, we plug Equation (3.2) and (3.3) into Equation (3.1), extract the common decision cost out of the summation, and come to:

\[
E(u^t|e^t) = -s(e^t) + p(e^t)r(e^t) + (1 - p(e^t))E(u^{t+1}|e^{t+1})
\]  

(3.4)

We recursively apply Equation (3.4) starting from the first lap and obtain:

\[
E(u^1|e) = \sum_{t=1}^{\infty} \left( \prod_{j=1}^{t-1} (1 - p(e^j)) \right) (-s(e^t) + p(e^t) r(e^t))
\]  

(3.5)

where \( e \) denotes the vector of all choices on the list. (The summation could alternatively be defined as a finite one if we assume a finite list but the derivation stays the same.)

Since the surplus captures all long-term benefits (via its reward part), \( u^1 \) in Equation (3.5) captures the surplus of the entire list. We explicitly wrote out the dependency of \( u^1 \) on all future cards (i.e. all following choices) by expanding “\( E(u^1|e^1) \)” to “\( E(u^1|e) \)” for the purpose of clarity.

Finally, from Equation (3.5), we follow the approach used in [9] by considering optimizing the order of each consecutive choice pair and obtain the IIR-PRP model: assuming that all \( p(e^t) \) are greater than 0, \( E(u^1|e) \) is max-
imized when the choices are ranked in decreasing order of:

\[ \rho(e^t) \overset{\text{def}}{=} r(e^t) - \frac{s(e^t)}{p(e^t)} \]  

(3.6)

We have thus mathematically demonstrated that the Interface Card Model is a generalization of the IIR-PRP model.

### 3.3 Navigational Card

We now come to the second instantiation example of the Interface Card Model, where we demonstrate that without assuming “sequential browsing” and given the availability of a richer set of navigational elements, the Interface Card Model can lead to very powerful optimization results that could not be achieved by any existing formal frameworks.

We go back to the classic IR setting where the user is looking for some items using the search engine, but we consider a new popular set of real world scenarios where we have some navigational elements to show on the interface in addition to the items themselves, which we collectively refer to as tags. For example, when we are searching for books in an online library catalog, we may use subject headings to quickly narrow down the set of books we need to examine. In a news browsing website, as another example, the news keywords could serve as navigational tags following which the user is able to identify an interesting news article much faster than they could if they are only given an article list, even a well optimized one. In general, these navigational tags themselves are not what the user is looking for, but they are linked to (possibly overlapping) subsets of items into which the user could quickly zoom by selecting the tags.

One key challenge in this setting is that, since the user is now faced with both a list of tags and a list of items, there is no longer a single list of choices which is assumed by [9], and the sequential browsing assumption no longer holds. As a result, many interesting questions regarding how to optimally generate a navigational interface in such cases cannot be answered...
in a theoretically rigorous way: e.g., (a) how many tags and items should we present in each interface instance, and how do we optimally partition the interface into the tag panel and the item panel? (b) should we allocate a larger proportion of the screen space to tags in smaller screens, and if so, how do we make such adjustment optimally? (c) along the interaction process, do we start by showing tags and then switch to the items when the system becomes more certain about the user’s information need, and if so, what would be the optimal time for the switch?

We address all these questions in a novel principled approach by establishing another instantiation of the Interface Card Model that only assumes a “sequential interaction” scheme without going further towards “sequential browsing”. We first present a set of assumptions and notations and then demonstrate the effectiveness of our approach.

**Definition 3.11** (Block). A block \( b \) is a display unit on a card representing either a tag or an item that could be selected by the user. A block representing a tag / item is referred to as a tag / item block.

**Assumption 3.5** (Navigational Card). Each card is a subset of blocks along with their presentation strategy.

The presentation strategy of the blocks on the card is a generalized notion that typically incorporates any ordering and/or panel layout of the blocks. Note that the user may or may not follow any order in examining the blocks (as is assumed in the traditional sequential browsing scheme).

**Assumption 3.6** (Selection Action). In each lap \( t \), the user could either select a block on the card \( q^t \) or select “next card”: \( A(q^t) = q^t \cup \{a^{t+1}_N\} \), where \( a^{t+1}_N \) denotes the “next card” action.

From now on, we will directly use \( q^t \) to designate the set of blocks on \( q^t \), and we use \( e \) to represent items (which we used to represent choices in the previous section). The “next card” action is a generalization of many real world user actions to skip everything shown in the current interaction lap and see more options, e.g. clicking “next page”, scrolling down, shifting eye.
focus one position down, etc.

**Definition 3.12** (Preference). The *preference* is the system’s estimated probability distribution characterizing the user’s interest in each item $e$. The system relies on the context to progressively update the preference along the interaction process and we designate the preference at lap $t$ by $p(e|c^t)$.

**Definition 3.13** (Item Action Model). The *item action model* for item $e$ is the user’s action model on the current card given their interest in item $e$: $p(a^{t+1}|e, q^t)$.

In practice, the item action model serves as the main linkage between our interface model and the item-tag relations. The intuition is that a user interested in a particular item would generally be more likely to select a tag related to the item. Of course, if the item block corresponding to the item itself is displayed on the card, the user would almost always select the item block rather than any tag block. But if neither the item block nor any related tag block is displayed, the user would most likely issue the “next card” action. We will come back to this in more detail later.

The original action model could now be written as the expected item action model, where the expectation is taken with respect to the preference:

$$p(a^{t+1}|c^t, q^t) = \sum_e p(e|c^t, q^t) p(a^{t+1}|e, c^t, q^t)$$

$$= \sum_e p(e|c^t) p(a^{t+1}|e, q^t)$$  \hspace{1cm} (3.7)

where we assume that (a) the preference is independent of the next card given the context and (b) the item action model is independent of the context given the item of interest to the user, both of which are very reasonable in general.

The rationale underlying such a decomposition of the original action model into two probabilistic models, the preference and the item action model, is two folds. Firstly, by dividing the action model at the item level, we allow for more flexibility in user modeling efforts in practice. Secondly, the decomposition naturally leads to a principled way of updating the preference.
via Bayes’ theorem:

\[
p(e|c^{t+1}) = p(e|c', q^t, a^{t+1}) \\
= \frac{p(e|c', q^t) p(a^{t+1}|e, c', q^t)}{p(a^{t+1}|c', q^t)} = \frac{p(e|c') p(a^{t+1}|e, q^t)}{p(a^{t+1}|c', q^t)}
\]

(3.8)

where \( p(a^{t+1}|c', q^t) \) comes from Equation (3.7) and we adopted the same two assumptions we made in deriving Equation (3.7).

To make the optimization problem more tractable, we make the following assumption about the reward function to prevent the optimization from depending on future laps:

**Assumption 3.7 (Information Gain Reward).** The reward of an action is the information gain in the preference distribution estimated in the next lap over the current lap:

\[
r(a^{t+1}|c', q^t) = InfoGain(p(e|c^{t+1}), p(e|c')) \\
= H(p(e|c')) - H(p(e|c^{t+1}))
\]

(3.9)

where \( p(e|c^{t+1}) \) comes from (3.8) and \( H \) is the information or entropy function: \( H(p) = -\sum p \log p \).

Intuitively, at a high level, the interactive retrieval process resembles an encoding of the user preference: the lower the entropy of the preference, the more the system knows about the user’s information need, and the easier it would be for the system to help the user find some interesting items. Therefore, the amount of reduction in the entropy of the preference becomes a natural choice for approximating the reward. (We could have explicitly written out the dependency of the reward on future laps just as what we did in the previous section, but it would make the computation overly complicated and intractable.)

Now, we come to address the problem that the user may not always follow the sequential browsing scheme while examining a card due to the fact that the card may often be more complicated than a simple list. Ideally, we want a “browsing model” to characterize the browsing behavior of users, which may
be obtained through user studies or estimated based on interaction logs, but as an initial step along such a generalization direction, we choose to focus on interfaces with a relatively small capacity with respect to humans’ attention, and we make the following two assumptions:

**Assumption 3.8 (Capacity Constraint).** The only constraint on the blocks shown on a card is that the total space the blocks occupy does not exceed the capacity of the card:

\[ f_c^t(q^t) = \sum_{b \in q^t} w(b) - 1 \quad (3.10) \]

where \( w(b) \) is the space block \( b \) occupies relative to the card.

**Assumption 3.9 (Uniform Cost).** The cost is assumed to be uniform across any action the user issues on a card, and only depends on the current context:

\[ s(a^{t+1}|c^t, q^t) = s(c^t), \forall a^{t+1} \in \mathcal{A}(q^t) \quad (3.11) \]

A key implication behind the capacity constraint is that, since it serves as the only constraint on the cards, we do not further impose any requirement regarding (a) what proportion of the card should be allocated to tag blocks and item blocks, (b) how many tag blocks and item blocks should be shown on the card, and (c) whether the card should be completely devoted to tag blocks or item blocks. Essentially, we are setting a completely “flexible” interface layout that our Interface Card Model could freely optimize. (We implicitly assumed that the blocks are all of regular shapes, so that a block could always be packed into the card as long as the amount of space left on the card is no less than the space the block occupies.)

Meanwhile, the uniform cost assumption also has a key implication: we assume the user could browse the blocks on the card in any order and, due to the relatively small capacity of the card, it always takes the user a constant, very small amount of attention to browse the blocks and make a decision on what action to issue next no matter what order the user follows in browsing the blocks. In such a way we are effectively relaxing the sequential browsing assumption.
With all the necessary assumptions and definitions laid down, we plug Equation (3.9), (3.10) and (3.11) into Equation (3.1). It is easily observed that two terms could be extracted out of the summation: (a) the entropy of the current preference, \( H(p(e|c')) \), and (b) the constant cost, \( s(c') \), and since these two terms do not involve \( q^t \), we could simply remove them from the objective function without affecting the optimization result. Eventually, the final optimization problem for our navigational card becomes:

\[
\begin{align*}
\text{minimize} & \quad \sum_{a^{t+1}\in A(q^t)} p(a^{t+1}|c',q^t) H(p(e|c^{t+1})) \\
\text{subject to} & \quad \sum_{b\in q^t} w(b) - 1 \leq 0 
\end{align*}
\]  

(3.12)

where \( p(a^{t+1}|c',q^t) \) and \( p(e|c^{t+1}) \) respectively come from Equation (3.7) and (3.8).

Now, we continue with analytical and real user experiments to demonstrate that Equation (3.12) leads to very interesting and powerful interface optimization results not achievable by any other existing method in principled ways.

Before we go into real computations, we first need to have (a) an initial preference model as the starting point for the series of context updates along the interaction, and (b) a working item action model. Since this study is not meant to be a user modeling study, from now on we simply assume a flat initial preference distribution:

**Assumption 3.10 (Uniform Initial Preference).** The initial user preference is uniform across a set of \( n \) items, i.e. \( p(e_i|c^1) = 1/n, \forall i = 1, 2, \ldots, n \).

In reality, the system usually has a better estimate of the user preference in the initial lap. For example, the a priori information may suggest to the system that the user is generally more interested in certain categories of items. Additionally, in cases of search engines, the system may have estimates of the probability of relevance for each item with respect to the user’s query, so that the probability of the user’s interest in each item along the ranked list returned by the system should be decreasing. The assumption of uniform
initial preference we make here is for the sake of computational convenience; it is solely meant to reduce some distracting details not relevant to the core model.

**Definition 3.14 (Item-Tag Map).** An *item-tag map* is a weighted bipartite network composed of (a) item nodes and tag nodes respectively corresponding to the set of all items and tags, (b) weighted edges between item and tag nodes if the item and tag they represent are related, with the edge weight quantifying the strength of their relation. A *uniform item-tag map* is an item-tag map in which all edges are of uniform weights. For nomenclature purpose, we say that a tag *covers* an item if there’s an edge linking their corresponding nodes in the item-tag map.

**Definition 3.15 (Simple Item Action Model).** Under the *simple item action model*, given the user’s interest in item $e$ and the set of blocks shown on card $q^t$, the user would issue an action based on the following three rules:

1. If the item block corresponding to $e$, $b_e$, is on the card, the user always selects it: if $b_e \in q^t$, then $p(b_e|e,q^t) = 1$; $p(b|e,q^t) = 0$, $\forall b \neq b_e$; and $p(a^{t+1}_N|e,q^t) = 0$.

2. Otherwise, if the card contains at least one tag block covering $e$, the user will either select one of these tag block(s) or “next card” with probabilities proportional to the corresponding edge weight(s) in the item-tag map and a predefined parameter $\varepsilon$, respectively:

\[
\begin{align*}
    p(b|e,q^t) &= \frac{v(e,b)}{\sum_{b' \in q^t} v(e,b') + \varepsilon} \\
    p(a^{t+1}_N|e,q^t) &= \frac{\varepsilon}{\sum_{b' \in q^t} v(e,b') + \varepsilon}
\end{align*}
\]

(3.13) (3.14)

where $v(e,b)$ denotes the weight of the edge between the nodes in the item-tag map representing $e$ and $b$.

3. Otherwise, the user will always select “next card”, i.e. $p(a^{t+1}_N|e,q^t) = 1$ and $p(b|e,q^t) = 0$, $\forall b \in q^t$.

The *simple uniform item action model* denotes the simple item action model
on top of a uniform item-tag map, and the \textit{perfect uniform item action model} is the simple uniform item action model with $\varepsilon$ set to 0.

We implicitly assumed in the second rule that, in cases of “competing” blocks, i.e. multiple blocks covering the same item simultaneously appearing on the card, the relative tendencies of the user selecting these blocks are kept constant, and equal the relative weights of the corresponding edges in the item-tag map. Such a simplification may not always hold in reality, since the relative tendencies of block selections might depend on the user, the lap, and other blocks on the card; however, it is in general a valid approximation and could greatly simplify the computation.

The user might sometimes accidentally miss a related tag and select “next card”, which could be captured using the $\varepsilon$ parameter, though we assume that the user would never miss the items they are interested in.

\subsection*{3.3.1 Analytical Experiments}

We apply our result for optimizing navigational cards in some simple example scenarios to analytically demonstrate the effectiveness of the Interface Card Model in generating optimal interactive interfaces. Although it might be possible to develop alternative ad hoc approaches that could result in the very same analytical solutions we derive here, our approach adopts a principled way that is solidly rooted in a theoretical IR model, which no existing approaches could achieve. In this section, we mainly focus on mathematically deriving the optimal conditions for the blocks on the card, and in particular the tag blocks (since the cases for item blocks are generally simpler); we leave the demonstration of our model’s effectiveness in automatically generating optimal interface layout in reality to the user study experiment.

To make the presentation cleaner, we omit the lap and context notions in all places: we assume that all the discussion here is about the optimization in the initial lap, and we adopt the \textit{uniform initial preference assumption}. We also adopt the \textit{perfect uniform item action model} for the sake of mathematical convenience. Furthermore, in order to better focus on the most crucial line
of the calculation without worrying about any trivial technical details, we assume a “perfect world” of tag navigation:

**Assumption 3.3.1.1 (Complete Tag Set).** There always exists some tag that precisely covers any given item subset.

As a consequence, we could entirely focus on deriving the mathematical conditions for the optimal tag(s) we should pick onto the card without worrying about whether such tag(s) actually exist or not in reality.

**One Tag Per Card**

In this example, we assume that the card only has space for a single tag block:

**Assumption 3.3.1.0.1 (One Tag Per Card).** \( w(b) = 1, \forall b. \)

The optimization question now becomes: what is the optimal number of items the picked tag should cover? If the user is interested in some item covered by the picked tag, then the user will select the tag; otherwise, the user will select “next card”. Based on Equation (3.8), in the first case, the preference is updated to narrow down towards the subset of items covered by the picked tag, and in the second case, the preference narrows down towards the subset of items not covered by the picked tag. Suppose the picked tag covers \( x \) items, \( x \in \{1, 2, \ldots, n\} \). We plug the entropies of the two updated preference distributions into Equation (3.12) and after some straightforward algebraic simplifications, the optimization problem becomes:

\[
\minimize_x \frac{1}{n} (x \log x + (n - x) \log (n - x)) \tag{3.15}
\]

We consider Equation (3.15) as a function of \( x \) and extend its domain to real numbers in \([1, n]\). By taking the derivative, we conclude that the minimization solution is:

\[
x = \frac{n}{2} \tag{3.16}
\]
Therefore, selecting a tag block covering around half of the items leads to an optimal card. This result shows the model tends to create a balanced partition of the item preference distribution, which coincides with our intuition.

Two Tags Per Card

In this example, we “shrink” the tag block and assume the card has space for two tag blocks:

**Assumption 3.3.1.0.2 (Two Tags Per Card).** \( w(b) = 1/2, \forall b. \)

Now, the optimization problem becomes two-folds: (a) how many items should each of the two picked tags cover? and (b) how many items should the two tags’ coverages overlap? To answer these two questions, let the number of items covered by the two tags respectively be \( x \) and \( y \), \( x, y \in \{1, 2, \ldots, n\} \), and let the number of common items covered by the two tags be \( t \), \( t \in \{0, 1, \ldots, n\} \), \( t \leq x, t \leq y, x + y - t \leq n \). A crucial difference between this example and the last one is that if the user is interested in some item in the two tags’ overlap, the user may select either one of them with equal probabilities, which affects the calculation of the action model and the updated preferences. After some tedious algebraic simplifications, the optimization problem in Equation (3.12) eventually comes to:

\[
\min_{x,y,t} \frac{1}{n} \left( t \log 2 + \left( x - \frac{t}{2} \right) \log \left( x - \frac{t}{2} \right) + \left( y - \frac{t}{2} \right) \log \left( y - \frac{t}{2} \right) + \left( n - x - y + t \right) \log \left( n - x - y + t \right) \right)
\]

(3.17)

Again, we consider Equation (3.17) to be a function of \( x \), \( y \) and \( t \), and we relax the integer constraint. By taking the partial derivatives, without going much into the technical details, we conclude that the final minimization solution is:

\[
x = y = \frac{n}{3}; \quad t = 0
\]

(3.18)
There are two implications from this result: (a) it reassures that the model would favor a balanced partition of the preference distribution, and (b) it additionally suggests that the model would minimize the partitions’ overlaps, coinciding again with our intuition.

3.3.2 User Study Experiments

To further demonstrate the effectiveness of our Interface Card Model, we built real prototype interface systems based on the navigational card model to show that our model could lead to automatic interface layout adjustment, which no existing method could achieve in a principled way, and we validate the superiority of our automatic interface layout results by comparing them with baseline pre-designed static interfaces in user studies.

The prototype interfaces were built on top of the set of most popular news articles and their associated keywords returned from the New York Times Most Popular API [59], in which the articles and the keywords respectively correspond to the items and the tags in our model. We developed two interfaces with different sizes, a medium sized one and a very small one. We assumed in our implementation that the user would follow the simple uniform item action model, and we heuristically set $\varepsilon = 0.5$.

Though the optimization problem in Equation (3.12) was shown to have closed form solutions in our two analytical experiments, it is generally difficult to solve for real world scenarios. For building the prototype interfaces, we implemented a straightforward randomized algorithm to tackle the problem. The algorithm heuristically generates multiple candidate cards at each lap, and chooses the one minimizing Equation (3.12). To obtain each candidate card, the algorithm picks blocks to add to the card one at a time that (a) do not violate the capacity constraint and (b) have a minimal overlap with all blocks that are already picked onto the card (as in line with what we observed in the analytical experiments).
Figure 3.1: Screen shots of example cards.
Sample Cards

In Figure 3.1, the left and top-right images are screenshots of an initial interface layout on the medium sized screen and the very small screen, respectively, as automatically determined by the Interface Card Model based on the popular news articles in New York Times and their keywords some time in late January 2015. We see that the algorithm intelligently decided to include only tag blocks on the small screen, but include both tag blocks and item blocks on the medium sized screen. Such a decision makes sense since unless we are relatively sure about what item the user is looking for (which unlikely happens in the initial interaction lap), it would likely be a waste of screen space if specific items are displayed; in contrast, tags are potentially more useful. The bottom-right screenshot in Figure 3.1 shows an automatic layout adjustment in response to the user’s action of selecting the “New York City” tag in the top-right interface. Despite its limited capacity, the screen is entirely filled with an item block because the estimated user preference is narrowed down to only a few items and the system determined that directly showing an item is more beneficial. These results demonstrate that our model can effectively achieve automatic layout adjustment according to both the screen size and the user interaction.

User Studies

We built two baseline interfaces for comparison purpose: one is for the medium sized screen, where we put a separate static tag panel on the right side of the main item panel; for the very small screen, we have either a tag panel or an item panel on the screen at each time, and put a switch button to allow users to switch between the two panels. These two baselines represent popular layouts seen on many mobile interfaces with medium and small screen sizes. We conducted real user experiments on Amazon Mechanical Turk [60] to compare the two baselines with our interfaces (on both medium and small screens) for a task of navigating into the most interesting article that was pre-identified by the user, and we measure the number of interaction laps for the users to reach their target article and compute p-values based on a one-side Wilcoxon sign-ranked test. We also varied the size of the item set
to see its impact. The results in Table 3.1 show that our interface outperforms the baseline interface in all the cases, though with varying significance levels (p-values less than 0.05 are highlighted). It is clearly observed that the superiority of our interface over the baseline interface is higher when the screen is smaller, and is also higher when there are more items.

Table 3.1: Significance levels of comparison results.

<table>
<thead>
<tr>
<th>Card size</th>
<th>Item set size</th>
<th>Valid sample size</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>20</td>
<td>19</td>
<td>0.004753</td>
</tr>
<tr>
<td>Small</td>
<td>50</td>
<td>23</td>
<td>0.0003546</td>
</tr>
<tr>
<td>Medium</td>
<td>20</td>
<td>18</td>
<td>0.09183</td>
</tr>
<tr>
<td>Medium</td>
<td>50</td>
<td>20</td>
<td>0.01097</td>
</tr>
</tbody>
</table>

We also asked the users survey questions for their opinions on the two types of interfaces to obtain some qualitative comparisons, and a majority of the users indicated that our interface was both quicker and easier to use. For example, one user wrote: “the interface seemed to intuitively know what article I wanted from just selecting two keywords.” Many users noted that the baseline interface felt familiar and thus was straightforward to use, but they also pointed out that it did not take much effort to lean how to use our interface: “at first I was unsure of how I would find my target article but followed my instincts and found it right away.” The difference in navigational efficiency between the two interfaces was more exaggerated in the very small screen, even in the search space of 20 articles. Since the baseline interface layout does not automatically switch between the keywords and the articles, a lot of users were not able to take full advantage of the keywords and simply ended up being scrolling through the entire article list: “it seemed like I had to search longer and scroll through almost every article to find the one I wanted.” In the medium sized screen, even though the baseline interface shows both the tag panel and the article panel, quite a few users noticed the ability of our interface to dynamically change the layout and applauded it: “I liked that the interface gave such large amount of results when you clicked on a tag.”
3.4 Summary

We proposed a novel general formal model for optimizing interactive information retrieval interfaces by viewing the interactive retrieval process as a process of a system playing a cooperative card game with a user with the goal of minimizing the user’s effort and maximizing the user’s gain of relevant information. At each interaction lap, the system would choose an optimal interface card (i.e., an interface instance) to present to the user based on the current context, a model of the user’s possible actions on the interface, and a model of the user’s gain and effort. The user can then choose an action to take on the prompted interface, which would lead to a new context for the system to choose the next optimal interface card.

We showed that this general Interface Card Model can cover the PRP for Interactive IR as a special case under a set of simplification assumptions, particularly the sequential browsing assumption (thus also easily cover the classic PRP as a more special case). We further derived a novel model for optimizing navigational interfaces that are adaptive to both the screen size and the user’s information need. Experimental results with real users show that the proposed model can effectively optimize a navigational interface and is significantly better than baseline static interfaces that are heuristically customized for different screen sizes.

The Interface Card Model is very general and can model interactions at any meaningful granularity level as long as we can define meaningful interface cards and user actions; thus we can model both “micro” interactions at the level of actions such as scrolling up/down inside a page, and “macro” interactions at the level of page navigation. The new model opens up many interesting new directions in optimizing the whole interactive retrieval system through incorporating machine learning and HCI study results. Specifically, the proposed formal framework naturally fits a wide variety of state-of-the-art machine learning techniques, and can easily adopt learning to rank methods [17, 18] and models such as the ostensive model [19] for evolving information needs to further improve the estimate of user preferences. With abundant interaction log data that can be recorded automatically, such learning techniques would provide more accurate estimate of multiple components in the
framework. Also, findings from HCI research could be directly incorporated into the constraint part in our optimization problem, providing our model guidance in certain domains that currently could not be formalized in a straightforward way, e.g. learnability concerns, error tolerance, etc. With the general trend in IR pushing researchers to focus more on the interface part and formalize interactive IR, we hope our work can stimulate alternative and more advanced formalism for interactive IR to be developed in the coming years (e.g., those in line of economic models for IR [20] and POMDP [23]).
The Interface Card Model (ICM) provides a high-level theoretical framework for optimizing interactive retrieval, but it does not specify a systematic way to instantiate it for solving a concrete interface optimization problem, leaving how to further refine this general framework an open challenge. We address this challenge and propose a novel formulation of the Interface Card Model based on sequential decision theory, which leads to a general instantiation of ICM that can explicitly model user states and stopping actions in the interactive retrieval process in a formal framework [13]. This formulation naturally connects the optimization of interactive retrieval with Markov Decision Process (MDP) and Partially Observable Markov Decision Process (POMDP) [61], thus enabling the use of reinforcement learning algorithms for optimizing interactive retrieval interfaces in general. We refer to the new model as Interface Card Model with User States (ICM-US).

While the proposed ICM-US model remains a high-level framework, it has several important advantages over ICM. Firstly, ICM-US explicitly models user states, which can potentially include many relevant variables about a user that we want to model (e.g., patience, attention tendency, and readability) in optimizing retrieval results. In this work, we particularly examine the modeling of stopping actions which is related to a user’s patience in search, and derive a framework to optimize the interface design with consideration of users’ stopping tendencies. Secondly, ICM-US opens up many opportunities to use sequential decision theories and reinforcement learning algorithms to optimize interactive retrieval. In this work, we show that it is possible to use the ICM-US framework to define and solve the interface optimization problems studied in the Interface Card Model in more elegant and more general ways. Specifically, we work out the “plain card” case in the sequential decision theory context and take the user stopping tendencies into consid-
eration, and mathematically prove that a more general ranking principle is
the solution to the Bellman Equation. In the “navigational card” case, we
consider user stopping tendencies and define a more general interface opti-
mization problem. The problem is NP-Hard and we conduct experiments
to tackle it in two ways: (a) we conduct simulated experiments to solve
the Bellman Equation under reasonable simplification assumptions; (b) we
approximately solve the optimization problem in more general settings and
conduct user studies to examine the empirical benefit of ICM-US. The results
demonstrate that ICM-US is effective for automatically adjusting the inter-
face layout in adaptation to inferred user stopping tendencies in addition to
user interaction and screen size.

4.1 Interface Card Model with User States

The original Interface Card Model is formalized as follows. Let $t$ be the
interaction lap under consideration, $q^t$ be an interface card the system could
issue to the user, $a^{t+1} \in A(q^t)$ be an action the user takes in the following lap
as a response to $q^t$, $^t$ and $f^t_c$ be the constraint function for $q^t$. Let $c^t$ be the
context accumulated till the current user action that starts from $c^1 = (i, a^1)$
(where $i$ is the prior information the system has about the user) and is
incrementally updated by the rule $c^{t+1} = (c^t, q^t, a^{t+1})$. Let $p(a^{t+1}|c^t, q^t)$ be
the user action model for characterizing how likely the user issues action $a^{t+1}$
given context $c^t$ and card $q^t$. Let $u^t$ be the estimated user surplus, which
equals the difference between the user’s reward and their cost for issuing an
action. Then the Interface Card optimization problem is defined as:

Definition 4.1 (Interface Card Optimization). In each lap $t$, the interface
system should play a card $q^t$ that maximizes the expected surplus $u^t$ given
the current context and under the current constraint, where the expectation

\footnote{As in the Interface Card Model, we assume the action set $A(q^t)$ is countable; the case
of uncountable action set could be handled via trivial changes to the model.}
is taken with respect to the user action model:

$$\max_{q^t} E(u^t|c^t, q^t)$$

$$= \sum_{a^{t+1} \in A^*(q^t)} p(a^{t+1}|c^t, q^t) u(a^{t+1}|c^t, q^t)$$

subject to $$f^t_c(q^t) \leq 0$$  \hspace{1cm} (4.1)

Instead of directly extending Equation (4.1) as in the Interface Card Model, we first introduce its intrinsic relation to sequential decision theories, and then redefine and expand the instantiations of the Interface Card Model in a more systematic way and derive new interesting results. In contrast to the typical practices adopted by other recent works in applying sequential decision theories in information retrieval (e.g., [23]), we are not deriving our framework based on sequential decision theories; all our derivation is self-contained and solely relies on the Interface Card Model and our own assumptions, and the formalisms in sequential decision theories we observe at the end are natural consequences of the derivation.

We first relate the interface optimizations in consecutive laps using the notion of context surplus:

**Definition 4.2** (Context Surplus). The *context surplus* is the maximum expected surplus across all possible cards $$q^t$$ subject to the constraint under the given context:

$$E(u^t|c^t) = \max_{q^t} E(u^t|c^t, q^t)$$

subject to $$f^t_c(q^t) \leq 0$$.

**Assumption 4.1** (Accumulative Surplus). The action surplus $$u(a^{t+1}|c^t, q^t)$$ takes the form of an arithmetic sum:

$$u(a^{t+1}|c^t, q^t) = u_0(c^t, q^t, a^{t+1}) + E(u^{t+1}|c^{t+1})$$  \hspace{1cm} (4.3)

The two components in the summation are: (a) $$u_0(c^t, q^t, a^{t+1})$$ - the *immediate action surplus* of action $$a^{t+1}$$ given card $$q^t$$ and under context $$c^t$$, which is the difference between the *immediate action reward* $$r_0(c^t, q^t, a^{t+1})$$ and the *immediate action cost* $$s_0(c^t, q^t, a^{t+1})$$; (b) $$E(u^{t+1}|c^{t+1})$$ - the context surplus
in the next lap.

Conceptually, we assume that the user obtains surplus in an accumulative fashion: it is usually reasonable in real world cases, which is also the reason why it became a standard practice in reward modeling in sequential decision theories. However, there might also be cases where the user surplus takes a non-additive form. For example, certain future reward could be conditioned on some particular card in advance, e.g. some instructive cards at the beginning to help the user better understand the interactive interface. In such cases, our assumption would become invalid and we would need to step back to the more general form of the original Interface Card Model; we leave it to future work.

There will always be a terminal lap in an interactive retrieval process, e.g. when the user fulfilled their information need or they could not find anything interesting and give up the interaction. To make our discussion more concise and modular, for the moment we assure that our proposed formulation could naturally characterize the case of terminal laps, and we will come back to this in Section 4.2.

In contrast to the diminishing reward model adopted in a large portion of sequential decision theories, where the future reward is multiplied by a discount factor \( \gamma < 1 \), we do not penalize future reward in the current most general form of our framework. Nevertheless, we will show in Section 4.2 that the diminishing reward model could be derived as a specialization of our framework to capture the user’s stopping actions in the interaction process, which gives the diminishing reward model a deeper and more principled reason of existence.

With Equation (4.2) and (4.3), we rewrite Equation (4.1) as:

**Definition 4.3** (Interface Card Bellman Equation). The expected surplus
in consecutive laps satisfies:

\[
E(u^t|c^t) = \max_{q^t} \sum_{a^{t+1} \in \mathcal{A}(q^t)} \left( p(a^{t+1}|c^t, q^t) \cdot 
(u_0(c^t, q^t, a^{t+1}) + E(u^{t+1}|c^{t+1})) \right) \tag{4.4}
\]

and the optimal card the system should pick in lap \( t \) is:

\[
q^t_{opt} = \arg\max_{q^t} \sum_{a^{t+1} \in \mathcal{A}(q^t)} \left( p(a^{t+1}|c^t, q^t) \cdot 
(u_0(c^t, q^t, a^{t+1}) + E(u^{t+1}|c^{t+1})) \right) \tag{4.5}
\]

with \( q^t \) in both equations subject to \( f^t_c(q^t) \leq 0 \).

It is often convenient to define:

\[
u_0(c^t, q^t) = \sum_{a^{t+1} \in \mathcal{A}(q^t)} p(a^{t+1}|c^t, q^t) u_0(c^t, q^t, a^{t+1}) \tag{4.6}\]

and transform Equation (4.4) to an equivalent form:

\[
E(u^t|c^t) = \max_{q^t} \left( u_0(c^t, q^t) + \sum_{a^{t+1} \in \mathcal{A}(q^t)} p(a^{t+1}|c^t, q^t) E(u^{t+1}|c^{t+1}) \right) \tag{4.7}
\]

Definition 4.3 forms the basis for all subsequent derivations in this chapter. Due to space limitations, we will only write out derivations for Equation (4.4) or its equivalent form Equation (4.7) in most places; the part for Equation (4.5) could be trivially filled out in each case.

Now we propose a fundamental assumption underlying all our following derivations regarding the modeling of the user:

**Assumption 4.2 (User State).** In each lap \( t \), the user lies in a unique, unambiguous *user state* \( z^t \) that contains all the necessary information to determine the user action model and the immediate action surplus when
given any interface card the system issues.

The user state is a general concept encapsulating many real world cases. For example, the user’s information need could be one straightforward type of user state, and there could also be more subtle user states such as binary states indicating whether the user is in the exploration or exploitation stage of their information seeking process; it is at the will of the practitioners of our framework to decide on the type of user states they would like to model. We will be define the user state $z^i$ in various forms in the following sections to fulfill diversified modeling needs.

The user state $z^i$ is closely related to the context $c^i$ in that they both characterize about the user for the system, but they are intrinsically different. There are two key insights we could obtain if we look back at Equation (4.4): (a) the user action model and the immediate action surplus are essentially all we need to fully carry out the computations in our framework; (b) it was the context that determined the user action model and the immediate action surplus when given the interface card, but its role is now fulfilled by the user state. More fundamentally, the context is always explicit to the system and contains all the information the system has at hand to speculate about the user, whereas the user state is intrinsic to the user and may or may not be known to the system, yet it is the essential information the system ever needs to know about the user. In other words, the user action model and the immediate action surplus are independent of the context given the user state - the user state serves as the sole linkage between the context and all the computational parts of our framework.

We first assume the user state is hidden from the interface system, and the interface system could only guess about the user state relying on the context.

**Assumption 4.3** (Initial User State Distribution). There exists an initial probability distribution over the user states, denoted by $d^1$, specifying how likely the user is in each user state at lap $t = 1$ that the system could estimate from the initial context $c^1$ about the user. We write $p(z^1|d^1) = p(z^1|c^1)$.

When the interaction starts, we assume that the user states may prob-
abilitistically transition from lap to lap when the user observes an interface card:

**Assumption 4.4 (User State Transition Function).** In each lap $t$, there exists a transition function $p_T^t(z^{t+1}|z^t, q^t)$ that specifies the probability of the user transitioning from $z^t$ to $z^{t+1}$ when the user observes $q^t$.

Now, starting from $p(z^1|d^1) = p(z^1|c^1)$, we could estimate the user state distribution $d^t$ inductively on $t$ based on Bayes’ rule:

$$p(z^{t+1}|d^{t+1}) = p(z^{t+1}|c^{t+1}) = p(z^{t+1}|c^t, a^{t+1})$$

$$= p(z^{t+1}|d^t, q^t, a^{t+1}) = \alpha p(a^{t+1}|z^{t+1}, q^t) p(z^{t+1}|d^t, q^t)$$

$$= \alpha p(a^{t+1}|z^{t+1}, q^t) \sum_{z^t} p(z^t|d^t) p_T^t(z^{t+1}|z^t, q^t)$$

(4.8)

where $\alpha^{-1} = p(a^{t+1}|d^t, q^t)$ is computed by:

$$p(a^{t+1}|d^t, q^t) = \sum_{z^t} p(a^{t+1}|z^{t+1}, q^t) p(z^{t+1}|d^t, q^t)$$

$$= \sum_{z^t} p(a^{t+1}|z^{t+1}, q^t) \sum_{z^t} p(z^t|d^t) p_T^t(z^{t+1}|z^t, q^t)$$

(4.9)

The context $c^t$ now provides no useful information additional to our estimated user state distribution $d^t$ for computational purposes, so it could essentially be replaced by $d^t$ in all places. If we define $u_0(d^t, q^t, a^{t+1})$ as:

$$u_0(d^t, q^t, a^{t+1}) = E(u_0|d^t, q^t, a^{t+1})$$

$$= \sum_{z^t} p(z^t|d^t) u_0(z^t, q^t, a^{t+1})$$

(4.10)

then Equation (4.4) becomes:

**Definition 4.4 (Interface Card Bellman Equation (Partially Observable...)**
User States)).

\[
E(u^t|d^t) = \max_{q^t} \sum_{a^{t+1} \in \mathcal{A}(q^t)} \left( p(a^{t+1}|d^t, q^t) \cdot (u_0(d^t, q^t, a^{t+1}) + E(u^{t+1}|d^{t+1})) \right)
\]

subject to \( f_c^t(q^t) \leq 0 \).

Equation (4.11) takes the exact same form as the standard Bellman Equation for Partially Observable Markov Decision Process (POMDP), where the user state \( z^t \), the user state distribution \( d^t \), the interface card \( q^t \) and the user action \( a^{t+1} \) respectively play the roles of state, belief state, action and observation (or often called evidence), and \( u_0(d^t, q^t, a^{t+1}) \) and \( E(u^t|d^t) \) respectively serve as the the reward function\(^2\) and the value function for belief states [61]; Equation (4.8) is the standard forward equation used for updating the belief state based on the action and the observation. In the language of POMDP, the state the interface system is interested in is the user state, and the actions the system could take are the interface cards it could issue to the user. However, since the user state is not fully observable, the system could only decide on the optimal action to perform according to the estimated distribution of user states, i.e. the belief state, and the estimation is computed based on the observations the system could collect about the user state, which are not surprisingly the user actions. We will see a concrete example of realizing Equation (4.11) to solve an interface optimization problem in Section 4.4.

Though the user states would usually be invisible to the system, we could sometimes assume that the system is actually aware of the user states for the sake of modeling convenience, either by explicitly asking or confirming about the user states, or by interpreting that the user actions (e.g. queries or clicks) have exact mappings to user states. In such cases, the belief state \( d^t \) is collapsed into an observable user state \( z^t \). With a series of simplifications, Equation (4.11) becomes:

**Definition 4.5** (Interface Card Bellman Equation (Fully Observable User

\(^2\)The reward function is defined on state-action-observation triples instead of on state-action pairs or just states as done in many studies which are special cases of our definition.
subject to \( f_c^t(q^t) \leq 0 \).

Equation (4.12) takes the exact same form as the standard Bellman Equation for Markov Decision Process (MDP), which is not a surprise since MDP could be derived as a special case of POMDP if observations uniquely determine states. We will see an example of realizing Equation (4.12) to solve a concrete interface optimization problem in Section 4.3.

4.2 Stopping Action Modeling

Definition 4.3 and its instantiations in the forms of Equation (4.11) and (4.12) have opened up enormous opportunities for research studies to apply the Interface Card Model to a very wide range of real world problems via tools from sequential decision and reinforcement learning theories. In this work, as an initial step, we will utilize our new framework to study a basic yet nontrivial aspect of interactive information retrieval - stopping actions. We claimed in Section 4.1 that our framework could naturally characterize such cases, and we now establish the formalism, starting by making the following assumption:

**Assumption 4.5 (User Stops Interaction).** The interactive process is always ended by the user - the system would always respond to user actions with optimized cards whereas the user may or may not choose to continue the interaction.

Our assumption applies to most real world scenarios where the user may choose to stop the interaction either because of satisfaction of their information need or frustration due to lack of useful information. There might
occasionally be cases where the system appears to be the one terminating
the interaction, e.g. if the system determines that nothing in its database is
interesting to the user and chooses to issue a “terminal card” that attempts
to stop the interaction with some possible explanations. In such cases, the
user is still free to choose between leaving the system and starting another
round of interaction, and indeed many would choose the latter, so we could
also model the termination as the user’s choice when facing the system’s
“terminal card”.

**Definition 4.6** (Stopping Action). In each lap \( t \), there is a stopping action \( a^{t+1}_B \in A(q^t) \) for any interface card \( q^t \), and its estimated probability under the current context \( c^t \), \( p(a^{t+1}_B|c^t, q^t) \), is called the stopping rate. The expected future surplus of the new context following the stopping action
\( E(u^{t+1}|c^{t+1}) = 0 \), where \( c^{t+1} = (c^t, q^t, a^{t+1}_B) \).

In the language of sequential decision theories, observing a user stopping
action is analogous to entering a terminal state (or belief state for POMDP)
for the system.

The stopping rate is typically dependent on the interface card the user
faces: e.g., it would be smaller if the user is interested in certain content on
the card, and larger if the content on the card does not look appealing to
the user for the past couple of laps. However, for the sake of modeling and
inference convenience, we may sometimes assume that the stopping rate is
constant:

**Assumption 4.6** (Constant Stopping Rate). The stopping rate is a constant
throughout the interaction: it is determined by the initial context \( c^1 \) and is
independent of the subsequent user actions and interface cards; we denote it
by: \( \beta^0 = p(a^{t+1}_B|c^1) = p(a^{t+1}_B|c^t, q^t), \forall c^t, q^t; \) and we restrict that \( 0 < \beta^0 < 1 \).

Assumption 4.6 is a “double-sided sword”: in the language of bias-variance
trade-off, it leads to a potentially large bias in exchange for less variance. In
this work, we will be cautious and make use of it only to our own advantage:
we will keep it for the rest of this section to derive one interesting theoretical
result and continue to rely on it in Section 4.3 to simplify the computation,
but we will discard it and resume the dependency of the stopping rate on the full context and the interface card in Section 4.4 for more realistic scenarios.

Now, with Assumption 4.6 in effect, if we define the user continuation action model as:

\[
p^K_t(a^{t+1}|c^t, q^t) = \frac{p(a^{t+1}|c^t, q^t)}{1 - p(a^{t+1}_B|c^t, q^t)} = \frac{p(a^{t+1}|c^t, q^t)}{1 - \beta^0} \tag{4.13}
\]

for all \(a^{t+1} \neq a^{t+1}_B\), then Equation (4.7) becomes:

**Definition 4.7** (Interface Card Bellman Equation (Constant Stopping Rate)).

\[
E(u^t|c^t) = \max_{q^t} \left( u_0(c^t, q^t) + (1 - \beta^0) \sum_{a^{t+1} \neq a^{t+1}_B} p^K_t(a^{t+1}|c^t, q^t) E(u^{t+1}|c^{t+1}) \right) \tag{4.14}
\]

subject to \(f^i_c(q^t) \leq 0\).

We could trivially extend Equation (4.14) to derive its instantiated forms for the MDP and POMDP cases based on Equation (4.12) and (4.11); we omit such derivations here, but we will be developing concrete models as examples of these two forms in Section 4.3 and 4.4, respectively.

More interestingly, the term \(1 - \beta^0\) in Equation (4.14) clearly resembles the discount factor \(\lambda\) for modeling diminishing reward in a form of the standard Bellman Equation for MDP and POMDP frequently seen in sequential decision theories (as well as many recent works in applying sequential decision theories in information retrieval, e.g., [23]), and the role of the (belief) state transition probabilities is here fulfilled by the user continuation action model. The resemblance is not a coincidence - it reveals a fundamental insight into the diminishing reward model in sequential decision theories. Traditionally, in addition to establishing an upper bound for the value function for mathematical conveniences, a major purpose for setting the discount factor is to express to the model our intuitive preference for quicker reward; in our case, we want to reduce the cost of excessive interaction laps for the user. In economic theories, every cost is intrinsically an opportunity cost, which
is formally defined to be the value of the best alternative [62]. In the setting of interactive retrieval, when the opportunity cost is higher than the expected surplus of carrying on the interaction longer, the user is very likely to switch to their best alternative way of spending their time, e.g. working on something else or simply relaxing, and from the system’s perspective, this is exactly the \textit{stopping action} of the user. Therefore, our model for user stopping actions turns out to draw fundamental connections from economic theories to sequential decision theories by providing deeper explanations to the diminishing reward model. We could look even further into discrete choice models [58] in modern economic studies to derive more rigorous formalisms for user decision modeling, and we leave it to future works.

4.3 Plain Card

As the first concrete extension of our proposed framework, we revisit the “plain card” setting in the Interface Card Model. We augment it with stopping actions, and formally re-define the optimization problem based on our new sequential decision formulation for Markov Decision Process (MDP) defined in Equation (4.12). We will mathematically prove that a more generalized ranking principle with user stopping tendencies taken into consideration is the solution to the Bellman Equation.

In the “plain card” setting in the Interface Card Model, the main assumption is that each interface card \( q^t \) is an atomic choice \( e^t \) placed on a ranked list and that the user examines and either accepts or rejects the choices in a sequential manner. Now, we additionally allow that the user may also stop after examining each choice. We continue to use \( p(e^t) \), \( r(e^t) \), \( s(e^t) \) to respectively denote the probability of the user’s interest in accepting \( e^t \), the immediate reward of accepting \( e^t \) and the immediate cost of examining \( e^t \) in lap \( t \), respectively, and these quantities are assumed to be independent of the past user actions as long as they have all been reject actions. An accept action is regarded as a “satisfying” stopping action - it terminates the current interaction session and starts a new one with an updated set of parameters. We now formally define this framework in the MDP language, starting with
defining the user state and the user action model:

**Definition 4.8** (Plain User State). The *plain user state* $z^t$ at lap $t$ is defined to be the set of choices not yet examined by the user till the previous lap: $z^t = \{ e : q^t \neq e, \forall t' < t \}$.

**Definition 4.9** (Plain User Action Model). The *plain user action model* for user state $z^t$ and choice $e^t$ is a distribution over the accept action $a_0^{t+1}$, the reject action $a_1^{t+1}$ and the stopping action $a_B^{t+1}$ defined under the constant stopping rate assumption as:

\[
\begin{align*}
    p(a_0^{t+1}|z^t, e^t) &= 1_{z^t}(e^t) p(e^t) \\
    p(a_1^{t+1}|z^t, e^t) &= 1 - \beta^0 - 1_{z^t}(e^t) p(e^t) \\
    p(a_B^{t+1}|z^t, e^t) &= \beta^0
\end{align*}
\]  

(4.15)

where $1_{z^t}(e^t)$ is the indicator function for testing whether $e^t \in z^t$, and we assume $p(e^t) \leq 1 - \beta^0$, $\forall e^t$.

Essentially, the plain user action model claims that the user will never accept a choice they have rejected before (as long as no accept action has taken place in the middle), which is very reasonable in real world situations.

For technical conveniences, we define the *choice surplus* of a choice $e$: $u(e) = p(e) r(e) - s(e)$, and we assume: (a) the set of all possible choices is finite, so the cardinality of $z^t$, $||z^t||$, is also finite; (b) if the choice shown on the interface is the only element left in $z^t$, then the user’s reject action is regarded as a “frustrating” stopping action; (c) $s(e) > 0$, $\forall e$, and (d) $u(e) \geq 0$, $\forall e$. Now, we could mathematically solve Equation (4.12) in closed form:

**Theorem 4.1** (Optimal Plain Card). Let $z^t \neq \emptyset$. Define:

\[
\eta(e) = \frac{u(e)}{p(e) + \beta^0} = \frac{p(e) r(e) - s(e)}{p(e) + \beta^0}, \quad e \in z^t
\]

(4.16)
Suppose $z^t = \{e_j\}_{j=1}^n = \{e_1, e_2, \ldots, e_n\}$ and $e_j$’s satisfy:

$$\eta(e_j) \geq \eta(e_k), \quad \forall \ 1 \leq j < k \leq n \quad (4.17)$$

Then $e_1$ is an optimal card for $z^t$ and:

$$E(u^t|z^t) = \sum_{j=1}^n \left( \prod_{k=1}^{j-1} (1 - \beta^0 - p(e_k)) \right) u(e_j) \quad (4.18)$$

Further, the complete optimization solution is to sequentially show $e \in z^t$ to the user in descending order of $\eta(e)$.

Proof. We prove by induction on $||z^t||$:

1. $||z^t|| = 1$. Let $z^t = \{e_1\}$, then:

$$E(u^t|z^t, e^t) = \begin{cases} u(e^t), & \text{if } e^t = e_1 \\ (1 - \beta^0)E(u^t|z^t) - s(e^t), & \text{otherwise} \end{cases} \quad (4.19)$$

Since $E(u^t|z^t) \geq E(u^t|z^t, e_1) = u(e_1) \geq 0$, we have $E(u^t|z^t, e^t) < (1 - \beta^0)E(u^t|z^t) \leq E(u^t|z^t), \quad \forall e^t \neq e_1$, i.e. showing any choice other than $e_1$ is non-optimal. So, $e_1$ is an optimal card for $z^t$ and $E(u^t|z^t) = u(e_1)$.

2. Suppose $n > 1$ and the theorem holds for all $z$ s.t. $||z|| = n - 1$. Suppose $z^t = \{e_j\}_{j=1}^n$ and $e_j$’s satisfies Equation (4.17). If $e^t \in z^t$, let $e^t = e_h, 1 \leq h \leq n$, then:

$$E(u^t|z^t, e_h) = u(e_h) + (1 - \beta^0 - p(e_h))E(u^{t+1}|z^{t+1}) \quad (4.20)$$

where $z^{t+1} = z^t \setminus \{e_h\}$. By induction hypothesis on $z^{t+1}$, we have $E(u^{t+1}|z^{t+1}) \geq 0$ from Equation (4.18). Thus, $E(u^t|z^t, e_h) \geq 0$, and $E(u^t|z^t) \geq 0$. Therefore, $E(u^t|z^t, e^t) = (1 - \beta^0)E(u^t|z^t) - s(e^t) < E(u^t|z^t), \quad \forall e^t \notin z^t$, i.e. showing any choice not in $z^t$ is again non-optimal.

$E(u^{t+1}|z^{t+1})$ in Equation (4.20) is by induction hypothesis maximized from ranking $e \in z^{t+1}$ in decreasing order of $\eta(e)$. Thus, if $e^t = e_h \neq e_1$,
then $e^{t+1} = e_1$. Let $z^{t+2} = z^t \setminus \{e_h, e_1\}$, $z^{t+1}_1 = z^t \setminus \{e_1\}$, then:

$$E(u^t|z^t, e_h) = u(e_h) + (1 - \beta^0 - p(e_h)) \cdot (u(e_1) + (1 - \beta^0 - p(e_1)) E(u^{t+2}|z^{t+2}))$$

$$\leq u(e_1) + (1 - \beta^0 - p(e_1)) \cdot (u(e_h) + (1 - \beta^0 - p(e_h)) E(u^{t+2}|z^{t+2}))$$

$$\leq u(e_1) + (1 - \beta^0 - p(e_1)) E(u^{t+1}|z^{t+1}_1)$$

$$= E(u^t|z^t, e_1)$$

(4.21)

where the first inequality comes from $\eta(e_1) \geq \eta(e_h)$ and the second inequality comes from $E(u^{t+1}|z^{t+1}_1) \geq E(u^{t+1}|z^{t+1}_1, e_h)$. Therefore, $e^t = e_1$ is optimal and $E(u^t|z^t) = E(u^t|z^t, e_1)$, which expands to the right hand side of Equation (4.18).

The definition for $\eta(e)$ in Equation (4.16) is identical to that of $\rho(e)$ in the Interface Card Model except an additional multiplier involving $\beta^0$:

$$\eta(e) = \frac{p(e)}{p(e) + \beta^0} \left( r(e) - \frac{s(e)}{p(e)} \right) = \left( 1 - \frac{\beta^0}{p(e) + \beta^0} \right) \rho(e)$$

(4.22)

If $\beta^0 \rightarrow 0$, i.e. when the user never abandons the search, then the two forms become equal. In the more general case where $\beta^0 > 0$, our new form of $\eta(e)$ may lead to a different ranking result where some choices with larger $p(e)$ values may be promoted because of the additional multiplier. Intuitively, when the user has a high tendency to stop, they would examine less choices on average, so by promoting choices with larger $p(e)$ values to higher places, the system could hope for a better chance of an accept action, and thus at least some reward, before the user leaves. Therefore, given a proper $\beta^0$ value (e.g. learned from user interaction logs), our new ranking principle defined by $\eta(e)$ could enable ranking with user stopping tendencies taken into consideration and form a basis for novel ranking algorithms on top of a richer user model.
4.4 Navigational Card

In this section, we revisit the “navigational card” setting in the Interface Card Model and formally re-define and solve the optimization problem based on our sequential decision formulation for Partially Observable Markov Decision Process (POMDP) defined in Equation (4.11). We will again incorporate stopping actions, but we will discard the constant stopping rate assumption and resume the more generalized and realistic setting: the stopping rate depends on the card and the full context. We will demonstrate that our new formalism leads to automatic interface adjustment based on users’ stopping tendencies in addition to the context and the screen size.

The “navigational card” setting in the Interface Card Model assumed the interface is backed by a set of information items denoted again by \( e \) (e.g. websites), each associated with some related tags (e.g. topics of websites), and the items and tags are respectively represented on the interface by item and tag blocks denoted by \( b \). A card \( q^t \) could contain any combination of item blocks and/or tag blocks, as long as the total area they occupy does not exceed the screen area: \( f^t_c(q^t) = \sum_{b \in q^t} w(b) - 1 \leq 0 \), where \( w(b) \) represents the space block \( b \) occupies relative to the entire screen size; the system may freely determine the layout of the interface by increasing or decreasing the number of item / tag blocks to display. Facing such a card \( q^t \), the user may either select a displayed block or issue the “next card” action \( a_{N}^{t+1} \) if nothing on the card interests them, the probabilities of which follow an item action model that depends on the user’s interest in them. The interface system estimates the user action model as an expectation of the item action model taken with respect to the estimated probability distribution of the user’s interest in each item based on the latest context.

In this work, we incorporate the stopping action \( a_{B}^{t+1} \) in all action set \( A(q^t) \). If the user finds an interesting item block and selects it, we consider it as a “satisfying” stopping action which triggers the interface system to jump to the item’s corresponding page. We define the user interest as a hidden user state and assume it doesn’t change across laps:

**Definition 4.10 (User Interest State).** A *user interest state* \( z^t \) denotes the
user’s interested item $e^t$ in lap $t$: $z^t = e^t$.

**Assumption 4.7** (Persistent User Interest). The user interest state does not change across laps: $p(e^{t+1}|e^t, q^t) = 1$ if $e^{t+1} = e^t$, 0 otherwise. From now on, we use $e^0$ to denote the user interest state.

Essentially, Definition 4.10 implicitly assumes that the user’s interest is focused on only one item within each lap, and Assumption 4.7 extends it to the whole interaction process. Both these two parts are sometimes inaccurate in reflecting real world scenarios, but they serve to exponentially reduce the complexity of our optimization problem.

With our simplification assumptions, Equation (4.8) and (4.9) respectively reduce to:

$$p(e^0|d^{t+1}) = \alpha p(a^{t+1}|e^0, q^t) p(e^0|d^t) \quad (4.23)$$

$$\alpha^{-1} = p(a^{t+1}|d^t, q^t) = \sum_{e^0} p(a^{t+1}|e^0, q^t) p(e^0|d^t) \quad (4.24)$$

In order to make the computations more tractable, we assume a constant immediate action cost $s^0$, a finite item set $E = \{e_1, e_2, \ldots, e_n\}$, and assume that we don’t have any useful prior information so we start with a flat belief state $d^1$: $p(e_j|d^1) = 1/n$, $1 \leq j \leq n$. We further make the following two common assumptions before separating into the two experiment sections:

**Assumption 4.8** (Uniform Item Reward). The reward to the user for selecting any item block in any lap is the same and is denoted by $r^0$.

Instead of modeling the actual expected reward of selecting an item block, $r^0$ could be regarded as measuring an eventual success in the interactive navigation, i.e. as the value of locating *any* interesting item to the user versus not finding anything at all.

**Assumption 4.9** (Simple User Interest Action Model). Let $v(e, b)$ be a measure of the intrinsic relation between item $e$ and block $b$. Let $\varepsilon \geq 0$, $0 < \delta < 1$. Given a user interest state $e^0$ and an interface card $q^t$, the user issues an action based on the following simple user interest action model:
1. If the item block for \( e^0, b^0_e \), is on \( q^t \) (i.e. \( b^0_e \in q^t \)), the user always selects it: 
\[
p(b^0_e | e^0, q^t) = 1, \quad p(b | e^0, q^t) = 0, \quad \forall b \in q^t, b \neq b^0_e, \quad \text{and} \quad p(a_N^{t+1} | e^0, q^t) = p(a_B^{t+1} | e^0, q^t) = 0.
\]

2. Otherwise, if \( q^t \) contains at least one tag block related to \( e^0 \) (i.e. \( \exists b \in q^t \) s.t. \( v(e^0, b) > 0 \)), the user will either select one of these tag block(s), select “next card” or stop: 
\[
p(b | e^0, q^t) = \alpha v(e^0, b), \quad \forall b \in q^t, \quad p(a_N^{t+1} | e^0, q^t) = \alpha \varepsilon (1 - \delta), \quad \text{and} \quad p(a_B^{t+1} | e^0, q^t) = \alpha \varepsilon \delta, \quad \text{where} \quad \alpha = \sum_{b \in q^t} v(e^0, b) + \varepsilon.
\]

3. Otherwise, the user either selects “next card” or stops: 
\[
p(a_N^{t+1} | e^0, q^t) = 1 - \delta, \quad p(a_B^{t+1} | e^0, q^t) = \delta, \quad \text{and} \quad p(b | e, q^t) = 0, \quad \forall b \in q^t.
\]

The uniform user interest action model denotes the simple user interest action model where each \( v(e, b) \) is either 1 or 0.

Conceptually, \( \varepsilon \) captures the chance the user misses to identify a related tag, and \( \delta \) captures the user’s stopping tendency given they have not identified any interesting block - either they missed one or there wasn’t any indeed. Despite the fact that \( \delta \) is treated as a constant, we are not assuming a constant stopping rate; the stopping rate is depending both on the card \( q^t \) and on the belief state \( d^t \). With sufficient user interaction log data, we may apply reinforcement learning algorithms to learn a more refined user action model in the real world. In this study, we assume the simple user interest action model to reduce the learning complication and focus more on the modeling part.

To apply our model in real problems, the only missing component now is the actual planning in the POMDP framework, and it is well known that planning in POMDP is in general NP-hard [61]. We reduce the planning problem to manageable forms via imposing additional assumptions on the cards in Section 4.4.1 and via employing planning heuristics in Section 4.4.2, and we leave explorations of general planning solutions to future work.
4.4.1 Simulation Experiments

In this section, we directly use the standard value iteration algorithm in sequential decision theories [61] to explicitly solve our interface optimization problem defined in Equation (4.11). We assume the user follows the uniform user interest action model, and we set $\varepsilon = 0$: the user never misses a related tag. Similar to what was assumed in the Interface Card Model, we also assume a complete tag set: $\forall$ item subset $E' \subseteq E$, $\exists$ block $b$ s.t. $v(e, b) = 1_{E'}(e)$, i.e. 1 if $e \in E'$, 0 otherwise. In addition, we make the following assumption on the card the interface system could issue in order to reduce to a linear belief state space:

**Assumption 4.4.1.1** (Exclusive Blocks). Any item $e$ is related to at most one block on any card $q^t$: $\forall e$, $\exists$ at most one $b \in q^t$ s.t. $v(e, b) = 1$.

Intuitively, this is a reasonable strategy for the system especially given that the system has access to a complete tag set: showing multiple blocks related to some item would not only confuse the user, but the system in turn would also be less precise in narrowing down into the item the user is truly interested in, ending up unnecessarily increasing the number of interaction laps.

More interestingly, this assumption guarantees that the belief state $d^t$ is always a uniform distribution over a subset of the items, which could be trivially reasoned via induction. Furthermore, due to the assumptions of uniform item reward and complete tag set, all belief states $d^t$ over the same number of items would appear identical to the system in terms of planning: any optimal policy for a belief state $d^t$ over e.g. items $\{e_1, e_2, e_3\}$ is completely reflective to one for a belief state $d^t$ over items $\{e_4, e_5, e_6\}$ or any other item subset of size 3. Therefore, the belief state space has now been reduced from a high-dimensional continuous space all the way to a one-dimensional discrete space of size $n$. From now on, if $d^t$ is over a set of $n$ items, we say the size of $d^t$, $||d^t||$, is $n$.

We conducted simulation experiments to employ the standard value iteration algorithm to solve the Bellman Equation via dynamic programming.
The experiments were performed for two screen sizes: a medium size (M) holding at most two item blocks or four tag blocks, and a small size (S) holding at most one item block or two tag blocks. We tested two values for $\delta$: 0.02 for simulating a relatively patient (P) user, and 0.2 for a relatively impatient (I) user. In total, we had four settings abbreviated as "MP", "MI", "SP", and "SI". We found in our experiment runs that varying $s^0$ and $r^0$ within reasonable ranges did not affect the experiment outcome in any fundamental way, and here we report the results we obtained when we set $s^0 = 1$ and $r^0 = 10$.

Figure 4.1 shows the value function of belief states as a function of their size. For all four settings, the value function decreases as the size of the belief state, i.e. their uncertainty level, increases, and the decreasing rates are all diminishing, implying that the interface system could reduce the belief state size in an exponential manner through interacting with the user. It is also clear that the value function is higher for medium screens than for small screens, and higher for patient users than for impatient users: a larger screen naturally helps the user navigate to their interested items in less laps, and a more patient user is more likely to stick to the interaction until they obtain the reward. Further, the difference between the value function for patient and impatient users is much smaller on the medium screen compared to that
on the small screen, suggesting that a larger screen is helpful in attracting the user to stick to the interaction by showing a wide variety of blocks to cater for the user’s interest.

Figure 4.2 shows the optimal policy in terms of the interface layout that our model determined for different belief state sizes. The interface is decided to be full of item blocks when the uncertainty of the belief state is low, and automatically changes to a combination of item and tag blocks and further to solely tag blocks as the uncertainty increases, which is a consequence of the tag blocks’ advantage in more quickly narrowing down the belief state size using less screen space as compared to the item blocks. It is not possible to display both item and tag blocks in the small screen, so the layout directly “jumps” from all item blocks to all tag blocks, and the “jump” also takes place when the belief state size is relatively small as compared to the case on medium screen, reflecting the more urgent need for more space-efficient tag blocks on smaller screens. More interestingly, the layout is also automatically adaptive to user stopping tendencies: when the user is less patient, our model intelligently adjusts the layout to show more tag blocks on both small and medium screens - with the hope of a better chance to hold the user onto the interaction when they see a tag related to their interest. Therefore, our proposed novel formulation of the Interface Card Model in the language of sequential decision theories successfully led to automatic interface layout
optimization results which could not only adapt to user interest and screen size, but also better cater for both patient and impatient users.

4.4.2 User Study Experiments

In this section, we apply our theoretical framework to solve interface optimization problems in real world settings. The assumptions we made in Section 4.4.1 would hardly exist in the real world (e.g. the complete tag set assumption), so we are again facing a general NP-Hard POMDP planning problem. We continue to assume that the user follows the uniform user interest action model, but in contrast to Section 4.4.1, we now allow \( \varepsilon > 0 \), which permits the possibility of the user missing to identify a related tag as is often the case in the real world.

Instead of using other sophisticated planning algorithms, we employ a straightforward and widely adopted heuristic in sequential decision theories, the dual-mode control heuristic [63], which picks actions (i.e. interface cards in our case) that lead to a minimal expected entropy value of the belief state. We slightly modify the heuristic to accommodate user stopping actions:

**Definition 4.11 (Entropy Heuristic (with stopping actions)).**

\[
q_{opt}^t = \arg\min_{q^t} \sum_{a^{t+1} \in A(q^t)} \left( p(a^{t+1}|d^t, q^t) \cdot \left( H(d^{t+1}) + 1_B(a^{t+1} r^0) \right) \right)
\]

subject to \( f_c^t(q^t) \leq 0. \) \((H(d^{t+1})\) denotes the entropy of \( d^{t+1} \) and \( 1_B(a^{t+1}) \) is shorthand for the indicator function of testing whether \( a^{t+1} = a_B^{t+1} \).)

The additional term \( 1_B(a^{t+1} r^0 \) we put into the dual-mode control heuristic is related to the eventual reward of finding an interesting item the user forfeited if they abandon the search (which we assumed to be uniform for all items and denoted by \( r^0 \)).

In order to assess the effectiveness of our proposed model in automatically
optimizing the interface layout of real interactive retrieval systems, we built prototype interface systems similar to those used in the Interface Card Model: we fetched popular news articles (as items) from the New York Times Most Popular API [59] together with their associated keywords (as tags), and we used Amazon Mechanical Turk (AMT) [60] to conduct user studies. We employed a straightforward randomized algorithm similar to the one used in the Interface Card Model to select the optimal interface card in each lap, but based on our new objective function defined in Equation (4.25). On the user side, we randomly partitioned the AMT workers into two groups, one being encouraged to stick to the interaction and thus playing the role of patient users, and the other being encouraged to freely give up the interaction and thus playing the role of impatient users; we refer to these two groups respectively as patient (P) and impatient (I) users. On the interface side, we varied the screen size and developed two sets of interfaces, one for a medium sized screen (M) being able to hold at most two item blocks or eight tag blocks, and the other for a very small screen (S) being able to hold at most one item block or four tag blocks. We again have four settings in total: “MP”, “MI”, “SP”, and “SI”.

We built two types of interfaces for each of the four settings for comparison. The first type is the baseline interface built based on the Interface Card Model (ICM) without user stopping tendencies taken into consideration - it is essentially always assuming a “perfectly patient” user. The second type is based on our new Interface Card Model with User States (ICM-US) with user stopping tendencies being considered, and it employs a straightforward learning method to infer users’ stopping tendencies. More specifically, in the uniform user interest action model, if we treat $\delta$ as the only variable we would make inference about and all other parameters as given, then the Maximum Likelihood Estimate (MLE) of $\delta$ is:

$$\delta_{MLE} = \frac{\text{#(stopping action)}}{\text{#(stopping action)} + \text{#(next page action)}}$$

(4.26)

where “#()” denotes the number of occurrences of the enclosed action in the interaction log. In our experiment, we estimated that $\delta_{MLE} = 0.029$ and 0.145 respectively for the group of patient and impatient users. Ideally, we could learn $\delta_{MLE}$ for each individual user, but due to the limited amount of
log data we could obtain, we decided to learn its value for each user group collectively. We noticed that our estimated \( \delta_{MLE} \) value for both user groups differed within \( \pm 8\% \) across the two screen sizes, implying that our uniform user interest action model is an adequately reasonable assumption for real world users.

We measure the effectiveness of the two types of interfaces using two metrics: (a) whether the users end up successfully finding an interesting article or not, referred as “success?”, and (b) how many laps the user spent for reaching an interesting article, referred as “#lap”. We use one-sided McNemar’s test for comparing “success?” and one-sided Wilcoxon sign-ranked test for comparing “#lap”. Table 4.1 shows the significance levels of our comparison tests for all four settings, where we use “•”, “⋆” and “⋆⋆” to represent a p-value within the range of (0.05, 0.1), (0.01, 0.05) and (0.001, 0.01), respectively\(^3\). It is clearly observed that our new interface is significantly better than the baseline interface at helping impatient users navigate to an interesting article without significantly increasing the number of laps they needed; the differences for patient users are not significant, which is also expected because the two types of interfaces would generate very similar optimization results due to a very small \( \delta \) value used in our new interface.

Table 4.1: Significance levels of comparison tests.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Sample size</th>
<th>“Success?”</th>
<th>“#lap”</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP</td>
<td>25</td>
<td>⋆⋆</td>
<td>⋆</td>
</tr>
<tr>
<td>MI</td>
<td>25</td>
<td>⋆⋆</td>
<td>⋆</td>
</tr>
<tr>
<td>SP</td>
<td>25</td>
<td>⋆⋆</td>
<td>⋆</td>
</tr>
<tr>
<td>SI</td>
<td>24</td>
<td>⋆</td>
<td>⋆</td>
</tr>
</tbody>
</table>

Interestingly, the differences in “success?” and ”#lap” are both less significant on the very small screen than on the medium sized screen for impatient users, contradicting to our expectation that our new model should benefit smaller screens more which are in nature less effective in keeping users engaged with the system. We speculate that, on the very small screen, both the baseline and the new interface would decide that the very limited screen space is more suitable for the “keyword layout” in most of the laps where

\(^3\)https://www.r-project.org/
the system is still very uncertain about the user’s interest, and the difference only occurs after the system could narrow down the user’s interest to within a few articles (as was also observed in Figure 4.2).

To give better intuitions into the interface optimization outcomes of our proposed model, we show some example screens we observed when we used our interfaces ourselves. Figure 4.3 and 4.4 display the second screen we observed on the very small screen respectively in the baseline interface and in our new interface (fed with the $\delta_{MLE}$ value estimated from the group of impatient users), after we click the “Colleges and Universities” keyword on the first screen (where both interfaces chose to display four popular keywords due to the very limited screen space). Given that there are three news articles associated with the clicked keyword, the baseline interface decided to adopt the “article layout” and display the three articles one by one to the user (of which the first one is shown in Figure 4.3), with the hope that the
user will thoroughly go over the three articles with the system. However, it is possible that an impatient user would stop if the first shown article does not interest them, and they would be even more likely to stop if they find the second shown article again not interesting. The new interface, on the other hand, considered user stopping tendencies in the optimization computation and thus chose to adopt the “keyword layout” and display three keywords within the same screen each corresponding to one of the three articles. Even an impatient user would likely identify a keyword related to their interest and continue on with the interaction by clicking it, and they would immediately reach their interested article in the next screen. It is also possible that the user might fail to recognize a related keyword, but the system determined that this would be a minor risk and would not affect the higher overall benefit of attracting an impatient user to stick longer to the interaction and eventually find something interesting. Note that all the reasoning above is only for our purpose to appreciate the intelligent layout decision of the interface system; the interface system is not built on top of any ad hoc logic to fulfill any part of our reasoning; its decision relies only on our optimization framework that elegantly captures all our intuitions in a single formulation.

Similar automatic layout decisions are observed on the medium sized screen as well, and the screen is even more capable in interface adaptations thanks to its additional intermediate layout choice of devoting half the space to an article and half to keywords. The results demonstrate the clear effectiveness of our proposed model in automatically optimizing interface layouts according to users’ stopping tendencies.

4.5 Summary

We proposed a novel refinement of the Interface Card Model based on sequential decision theory, i.e. ICM-US, that can facilitate formal user modeling and naturally connect optimization of interactive retrieval with Markov Decision Process and Reinforcement Learning (RL) in a general way, thus enabling the use of RL to solve potentially a wide range of problems of optimal interface design. ICM-US opens up many opportunities for formally modeling
user behavior in interactive retrieval as well as incorporating user behavior models into an optimal retrieval algorithm, making the retrieval algorithm “sensitive” to user behavior.

An obvious future direction is to further explore the large space of specific refinements of ICM-US and apply ICM-US to many applications to optimize interactive search and interface design. Another very interesting direction is to use ICM-US to analyze interactive search log data for discovery of interesting user behavior patterns or testing hypotheses about users’ search behavior.
CHAPTER 5

BAYESIAN FRAMEWORK FOR USER PREFERENCE IN INTERACTIVE INFORMATION RETRIEVAL

5.1 A Bayesian Probabilistic Framework

As an example of applying the Interface Card Model in optimizing a larger scale real world interactive retrieval system, we focus on the user-system interactions in a faceted browsing system and study how to model the user’s preference in each lap of the interaction process and how to optimize the system accordingly [14]. In a faceted browsing system, the user may first issue a natural language query to the system, and then typically select one or multiple facets to “zoom in” to the specific set of products the user is interested in. We propose that before the user selects each facet filter, the system has a prior estimate of the user’s interest in each item (e.g. based on the relevance score of each item with respect to the user’s query and the user’s previously selected filters), and when the user selects a facet filter, the system computes a posterior estimate of the user’s interest in each item based on the prior estimate and the user’s selected facet filter. Intuitively, the system’s posterior estimate of the user’s interest in an item is dependent upon (a) the prior estimate, and (b) whether the user would be likely to pick the selected facet filter if the user is in fact interested in this item. In other words, the user’s facet filter selection action serves as an additional supporting “hint” to the system for probabilistically re-inferring the user’s interest. We now formally cast all such intuitions into a principled Bayesian probabilistic framework:

**Definition 5.1** (Bayesian preference update). When observing the user’s selection of facet filter $a$, the system’s estimate of the user’s posterior propensity $p(e|a)$ in each product $e$ is derived from the user’s prior propensity $p(e)$ and
the user's action model \( p(a|e) \) via Bayes' theorem:

\[
p(e|a) \propto p(e) p(a|e)
\]  

(5.1)

According to this Bayesian probabilistic framework, the system’s estimate of the user’s preference after the filter selection depends on two components: the prior propensity \( p(e) \) and the user action model \( p(a|e) \). The prior propensity characterizes the system’s belief before the filter selection on how likely the user is interested in each item. We do not discuss in detail about the initial prior propensity in this work, i.e. the prior propensity before the user selects any facet filters. In practice, the initial prior propensity could be estimated in different ways, e.g. from an initial relevance ranking as well as additional personalization information of the user if available, and is generally available to us in a search system which assesses the relevance of items based on probability of relevance.

The Bayesian probabilistic framework could be applied iteratively in cases where the user selects multiple facet filters: the posterior propensity would serve as the prior propensity for the next user action. Such a desirable property is a natural consequence of the Bayesian formalism theory.

The action model characterizes how likely the user selects a filter given their interest in a particular item. If we restrict \( p(a|e) \) so that it equals 1 if and only if \( e \) satisfies \( a \), and 0 otherwise, then it is easily observed that our Bayesian probabilistic framework is reduced to the traditional “hard” faceted browsing scheme, where only the items satisfying the applied filter are returned and ranked in their original order. Such system behavior, which we will phrase as traditional filtering or “hard” filtering, might not be ideal in many cases. For example, when a user selects a price range as a filter, the user might also be interested in some products that are priced slightly higher than the range but are at a significant discount; when a user selects a brand filter, the user might also be interested in products of a very similar brand. In our proposed Bayesian framework, if we allow \( p(a|e) \) to be a true probabilistic distribution rather than concentrating on a single \( a \), as in the more general case, we are essentially capturing the uncertainty underlying user actions and would thus lead to smarter interactions between the user...
and the system. We will use concrete examples to better illustrate this in the following sections.

The action model in a more general setting can probabilistically characterize a much wider range of user actions in addition to facet filter selections, such as query reformulation, conversational interactions with the system, etc. The Bayesian probabilistic framework we propose here could serve to provide formal guidance in such scenarios, thus opening up many interesting directions for future research.

5.2 Action Model and Inference

In this section, we discuss concrete ways to instantiate the probabilistic distribution underlying the action model in the cases of some example facets in e-commerce search systems, and we introduce Bayesian inference methodologies for parameter estimation for each probabilistic distribution based on user search log. The example facets are representative of the major types of facets, and the e-commerce search engine is a typical example of a faceted browsing system. Thus, the techniques discussed here are universal and could generally be applied to other facets and other faceted browsing systems.

5.2.1 Categorical Filter

In a typical faceted browsing system, a lot of facets take values from an unordered set of values. One example is the brand facet in an e-commerce search system, where each product has a brand value from the set of possible brand names. In reality, the brand of the products users are interested in may often be different from the brand they select as filters when they are interacting with the system. Such phenomena are naturally due to the fact that some brands are similar to each other, so that the users selecting one brand might also be interested in some products of another brand. In this section, we will use brand filter as an example of unordered filter and demonstrate how we apply our proposed Bayesian framework to optimize the
system’s interaction.

The action model in the case of brand, \( p(b|e) \), would capture how likely a user would select each brand filter \( b \) given the user’s interest in a product \( e \), and the most natural choice for such an action model is the categorical distribution.\(^1\) Specifically, suppose there are altogether \( k \) brands: \( b_{(1)}, b_{(2)}, \ldots, b_{(k)} \). Then for each product \( e \), there exists a probability vector \( \mathbf{p}_e = (p_{e(1)}, p_{e(2)}, \ldots, p_{e(k)}) \) such that:

\[
p(b_{(i)}|e) = p_{e(i)}, \quad i = 1, 2, \ldots, k
\]

Note that if \( b_{(i*)} \) is the brand of \( e \), then the probability \( p_{e(i*)} \) should typically be the highest among all \( p_{e(i)} \)'s, and the probabilities corresponding to brands similar to \( b_{(i*)} \) should generally be higher than those corresponding to more distant brands. (In the extreme case where we set \( p_{e(i*)} \) to be 1 and all other \( p_{e(i)} \)'s to be zero, it could be easily observed that our model reduces to the traditional “hard” filter.)

To estimate \( \mathbf{p}_e \) for a product \( e \), we rely on the conjugacy relationship between the Dirichlet distribution and the categorical distribution \([64]\). In particular, let the prior distribution of \( \mathbf{p}_e \) be:

\[
\mathbf{p}_e \sim \text{DIR}(\alpha_e^{(0)})
\]

where \( \text{DIR}(\alpha_e^{(0)}) \) is a Dirichlet prior with the hyper-parameter vector \( \alpha_e^{(0)} = (\alpha_{e(1)}^{(0)}, \alpha_{e(2)}^{(0)}, \ldots, \alpha_{e(k)}^{(0)}) \). \(^2\) The hyper-parameters \( \alpha_{e(i)}^{(0)} \)'s, in practice, should be set to reflect the prior belief regarding the degrees to which each brand is related to the product. For example, in the most naïve scenario, if \( b_{(i*)} \) is the brand of \( e \), then \( \alpha_{e(i*)}^{(0)} \) could be set to be a non-zero value and all other \( \alpha_{e(i)}^{(0)} \)'s are set to be zero or a very small valued (for Laplace smoothing).

Next, we make inference by collecting from the user search log the brand filters selected in all search sessions leading to an eventual purchase (or “add-to-cart action”) of \( e \), and we denote the vector of these brand filters by

\(^1\)Since we are characterizing individual brand filter selection actions rather than multiple selection actions as a whole, we use the categorical distribution instead of the multinomial distribution.

\(^2\)The superscript “(0)” is used to label the prior hyper-parameters.
\( \mathbf{b}_e^{(n)} = (b_1, b_2, \ldots, b_n) \). Then, the posterior distribution of \( \mathbf{p}_e \) could be derived from its prior distribution and observations \( \mathbf{b}_e^{(n)} \) based on Bayes’ theorem:

\[
p(\mathbf{p}_e | \mathbf{b}_e^{(n)}) \propto p(\mathbf{p}_e) \ p(\mathbf{b}_e^{(n)} | \mathbf{p}_e) \tag{5.4}
\]

Due to the property of conjugacy, the posterior also takes the form of a Dirichlet distribution:

\[
\mathbf{p}_e | \mathbf{b}_e^{(n)} \sim \text{DIR}(\alpha_e^{(n)}) \tag{5.5}
\]

where \( \alpha_e^{(n)} = (\alpha_e^{(n)(1)}, \alpha_e^{(n)(2)}, \ldots, \alpha_e^{(n)(k)}) \) is the hyper-parameter vector in the Dirichlet posterior that is updated from \( \alpha_e^{(0)} \) according to:

\[
\alpha_e^{(n)(i)} = \alpha_e^{(0)(i)} + \sum_{j=1}^{n} \mathbb{1}\{b_j = b_{e(i)}\}, \quad i = 1, 2, \ldots, k \tag{5.6}
\]

where “\( \mathbb{1} \)” is the identity function that takes value 1 if the condition is satisfied and 0 otherwise. Note that such a posterior update procedure could continue on and on when new observations are obtained from the search log due to the property of conjugacy.

The posterior estimate of \( \mathbf{p}_e \) could either be derived from a maximum a posteriori point estimate from the posterior distribution, or from a posterior predictive distribution. In the case of the Dirichlet-Categorical conjugacy, these two alternative methods lead to the identical estimates:

\[
\hat{\mathbf{p}}_e^{(i)} = \frac{\alpha_e^{(n)(i)}}{\sum_{i'=1}^{k} \alpha_e^{(n)(i')}} \), \quad i = 1, 2, \ldots, k \tag{5.7}
\]

### 5.2.2 Ordinal Filter

In contrast to unordered facet value sets, facet values in many cases are ordinal, where there is a strict total ordering within the set of all possible facet values. The categorical distribution in such a scenario is unable to capture the ordinal relationships among the filters. In an e-commerce search

---

\(^3\)The superscript “\((n)\)” is used to label the observation vector of size \( n \) as well as the posterior hyper-parameters estimated after seeing the observations.
engine, the price facet is one of many examples. The filters corresponding to the ordinal facets are often presented in ranges, e.g. "$150-$200". It is observed that when a user selects a particular price range filter, they tend to be more interested in some products priced at around the middle of the range rather than products with prices near the boundary, and there is some slight chance that they might be interested in some products priced completely outside the range, e.g. some product just a little above the range yet having a substantial value relative to its price. In this section, we will use price range filter as an example of ordinal filter to demonstrate how we could apply our proposed Bayesian framework to optimize the system’s interaction.

We employ the Gaussian distribution to derive the action model for price facet $p(r|e)$ - the probability of a user interested in a product $e$ selecting a price range $r$. We postulate that given the user is interested in a product $e$, the probability density of the user selecting a particular price value $c$ follows a Gaussian distribution: $c|e \sim \mathcal{N}(\mu_e, \sigma^2_e)$, and the cumulative probability of the user selecting a particular price range filter $r = [a_r, b_r]$ is computed via integration of the Gaussian density function:

$$p(r|e) = \Phi \left( \frac{b_r - \mu_e}{\sigma_e} \right) - \Phi \left( \frac{a_r - \mu_e}{\sigma_e} \right)$$

where $\Phi$ denotes the cumulative distribution function of the standard Gaussian distribution. Note that $\mu_e$ should typically be close to the price of $e$, $c_e$, but users may not have a precise idea of the price of their interested product, so we treat $\mu_e$ as unknown and learn its value from user search log. Under such a Gaussian model for price filter selection actions, given that a user is interested in a particular product with a price tag $c$, they may most likely select a price range filter that covers $c$ at around the mid-point of the range, and would less likely select a price range filter with its boundary very close to $c$, and it would be even less likely but not impossible that they select some price range filter not covering $c$ at all. Such consequences nicely coincide with our intuition. (In the extreme case where we set $\mu_e = c_e$ and $\sigma^2_e \to 0$, it could be easily observed that our model reduces to the traditional “hard” filter.)

The two parameters $\mu_e$ and $\sigma^2_e$ are typically unknown in the real world, so we need to make inference from observations of past user activities. In
Bayesian statistics theories, the Gaussian distribution with both its mean and variance unknown has the Normal-Inverse-Gamma (NIG) distribution as its conjugate prior. Thus, we define the prior distribution for \( \mu_e \) and \( \sigma_e^2 \) as:

\[
\mu_e, \sigma_e^2 \sim \text{NIG}(\mathcal{H}_e^{(0)})
\]

\( \mathcal{H}_e^{(0)} = (\mu_e^{(0)}, \kappa_e^{(0)}, \alpha_e^{(0)}, \beta_e^{(0)}) \) represents the hyper-parameter vector in the NIG prior in the form of:

\[
\begin{aligned}
\sigma_e^2 | \alpha_e^{(0)}, \beta_e^{(0)} &\sim \text{IG}(\alpha_e^{(0)}, \beta_e^{(0)}) \\
\mu_e | \sigma_e^2, \mu_e^{(0)}, \kappa_e^{(0)} &\sim \mathcal{N}(\mu_e^{(0)}, \sigma_e^2/\kappa_e^{(0)})
\end{aligned}
\]

where “\( \text{IG} \)” denotes the inverse gamma distribution. In practice, the hyper-parameters could be heuristically set based on any available prior knowledge about how users select price ranges. For example, we typically set \( \mu_e^{(0)} = c_e \).

For each product \( e \), we collect the price filters selected in all the sessions that resulted in an eventual purchase (or “add-to-cart action”) of \( e \) from the search log and form the set of observations for making inference on \( \mu_e \) and \( \sigma_e^2 \). To make the inference computation tractable, we pick the mid point \( m_r \) of the price range in each selected filter \( r \) as an approximation to the whole range.

Thus, for each product \( e \), we obtain an observation vector composed of the mid points of all the selected price filters for product \( e \), and we denote the vector by \( \mathbf{m}_e^{(n)} = (m_1, m_2, \ldots m_n) \). Following that, the posterior distribution for \( \mu_e \) and \( \sigma_e^2 \) could be derived from their prior distribution and observations \( \mathbf{m}_e^{(n)} \) based on Bayes’ theorem:

\[
p(\mu_e, \sigma_e^2 | \mathbf{m}_e^{(n)}) \propto p(\mu_e, \sigma_e^2) p(\mathbf{m}_e^{(n)} | \mu_e, \sigma_e^2)
\]

Due to the property of conjugacy, the posterior also takes the form of an NIG distribution:

\[
\mu_e, \sigma_e^2 | \mathbf{m}_e^{(n)} \sim \text{NIG}(\mathcal{H}_e^{(n)})
\]

where \( \mathcal{H}_e^{(n)} = (\mu_e^{(n)}, \kappa_e^{(n)}, \alpha_e^{(n)}, \beta_e^{(n)}) \) is the hyper-parameter vector in the NIG posterior that is updated from \( \mathcal{H}_e^{(0)} \) based on the sample mean \( \bar{m}_e \) and vari-

\[\text{4 Since the price ranges of the price filters in e-commerce search engines are often relatively short segments as compared to the magnitude of product prices, it is typically reasonable to approximate the whole ranges by their mid points.}\]
Note again that such a posterior update procedure could continue on and on when new observations are obtained from the search log, due to the property of conjugacy.

With the posterior distribution of \( \mu_e \) and \( \sigma_e^2 \) established, two alternative methods could be used for estimating the Gaussian model for price filter selection actions. The first one is directly based on the maximum a posteriori point estimate from the posterior:

\[
\hat{p}(r|e) = \Phi \left( (b_r - \hat{\mu}_e)/\hat{\sigma}_e \right) - \Phi \left( (a_r - \hat{\mu}_e)/\hat{\sigma}_e \right) \tag{5.18}
\]

where \( \hat{\mu}_e \) and \( \hat{\sigma}_e^2 \) could be shown to come from:

\[
\begin{cases}
\hat{\mu}_e = \mu_e^{(n)} \\
\hat{\sigma}_e^2 = \beta_e^{(n)}/(\alpha_e^{(n)} + 3/2)
\end{cases} \tag{5.19}
\]

The second estimation method comes from the posterior predictive distribution, which, in the case of Normal-Inverse-Gamma distribution, could be shown to follow a \( t \)-distribution:

\[
\tilde{p}(r|e) = \Psi \left( (b_r - \tilde{\mu}_e)/\tilde{\sigma}_e \right) - \Psi \left( (a_r - \tilde{\mu}_e)/\tilde{\sigma}_e \right) \tag{5.21}
\]

where \( \Psi \) denotes the cumulative distribution function of a \( t \)-distribution with \( 2\alpha_e^{(n)} \) degrees of freedom, and \( \tilde{\mu}_e, \tilde{\sigma}_e^2 \) come from:

\[
\begin{cases}
\tilde{\mu}_e = \mu_e^{(n)} \\
\tilde{\sigma}_e^2 = \beta_e^{(n)} \left( \kappa_e^{(n)} + 1 \right)/\left( \alpha_e^{(n)} \kappa_e^{(n)} \right)
\end{cases} \tag{5.22}
\]
In practice, these two methods lead to almost identical inference results given a moderately large observation vector. In our experiments, we always employ the first method.

5.3 Experiments

To demonstrate the effectiveness of our proposed model, we implemented our Bayesian probabilistic framework in an internal prototype system on top of the Walmart e-commerce search engine. In the experiments, for each product $e$, we collect the price filters selected in all the sessions that resulted in an eventual purchase of $e$ from the search log and form the set of observations for making inference on $\mu_e$ and $\sigma^2_e$. To make the inference computation tractable, we pick the mid point $m_r$ of the price range in each selected filter $r$ as an approximation to the whole range.

We performed extensive experiments using search log data to compare our model with a baseline “hard” filtering model in which the system always returns the set of products strictly priced within the selected price filter, as performed in most standard faceted browsing systems. We collected around 62,000 search sessions in a month in 2014 in which the user (a) selected at least one price filter (which was implemented in the manner of traditional filters) and (b) purchased one product at the end of the session. Then, we carried out simulated user evaluations relying on this search log dataset. We collected 20 most popular queries that led to at least 700 purchases in the month, and we performed leave-one-out cross validations to compare our proposed model with the traditional faceted browsing scheme via simulated user interactions with the system. For each query, among all filters ever selected by any user who issued the query and eventually made a purchase, we treated one of them as the test data and all the rest as the training data at a time. Specifically, we trained our model using all but one of the filters to learn the user action model for all the products returned by the search engine, and then applied the trained model to the filter that was left as the test data and recorded down the rank of the user’s eventually purchased product. Meanwhile, we also applied the traditional filter to all the products returned by the
search engine, and recorded the rank of the user’s eventually purchased item. In cases where the traditional filter missed the user’s eventually purchased item, we tried not to “over-penalize” the traditional filter and computed the rank as the total number of filtered products plus the rank of the user’s purchased product in the original unfiltered list, emulating the scenario where the user scans the entire filtered list without finding the product, de-selects the filter, and then scans the original list to look for the product. We performed one-sided Wilcoxon signed-rank tests for the comparison for each of the 20 queries, and we heuristically set the prior parameters for our model. We observe that 19 out of the 20 queries have a p-value less than $10^{-6}$, strongly indicating that our soft faceted browsing scheme is significantly superior than the traditional “hard” faceted browsing scheme in terms of its efficiency in helping users navigate to the products of their interests. The only one exception is the query “electronics” with p-value 0.00187 (which is also significant though not as extreme as the other queries), and the reason is observed to be that most of the users’ purchases concentrated on the very top portion of the ranked list returned by the search engine and that most of the users’ issued filters happened to cover these items, in which cases the traditional filter is relatively harder to beat.

5.4 Summary

We proposed a novel soft faceted browsing scheme for information access systems as an example of applying the Interface Card Model in optimizing a larger scale real world interactive retrieval system where, when the user selects a facet filter, the system may return a few relevant items that do not satisfy the filter in a non-intrusive way alongside the items that satisfy the filter. We provided a formal Bayesian probabilistic framework for realizing such a soft faceted browsing scheme that takes into consideration the probabilistic nature of users’ facet filter selection actions, and we demonstrated that our method is more effective than traditional “hard” faceted browsing scheme via experiments using e-commerce search log data. The proposed framework and model also opens up interesting new research opportunities in the intersection of machine learning and information retrieval. An interesting extension is to
introduce active learning for optimal preference elicitation (e.g., dynamically adjust the price ranges to focus on the most uncertain range of prices).
CHAPTER 6

SEARCH SIMULATION FRAMEWORK FOR EVALUATING INTERACTIVE INFORMATION RETRIEVAL

Information Retrieval (IR) is an empirically defined task in the sense that there is no way to mathematically prove one IR system is better than another, and the question of which IR system is the best can only be answered based on how well the system can help users finish a task. Thus, how to appropriately evaluate an information retrieval (IR) system has always been one of the most important research questions in IR [39, 40, 41]. So far, the dominant methodology for evaluating an IR system has been the Cranfield evaluation methodology proposed in 1960s [43]. The basic idea is to build a test collection that consists of a sample of queries, a sample of documents, and a set of relevance judgments (indicating which documents are relevant/non-relevant to which queries). An IR system can then be evaluated using such a test collection as follows. First, we run the system on the test collection to generate retrieval results for each of the test queries. We then quantitatively evaluate the system results for each query with various measures (such as precision and recall) based on the relevance judgments. Measures on all the queries can be aggregated to quantify the performance of a system on the whole set of queries. Such a methodology has also been widely used for evaluating many other empirical tasks, including particularly machine learning tasks.

A key benefit of using the Cranfield evaluation methodology is that the test collection, once built, would be reusable as many times as we want to, which enables repeatedly using the same test collection to compare different systems or examine the effectiveness of each component in a complicated system. Such reusability is key to ensure reproducibility of IR experiments. The Cranfield evaluation methodology has played a crucial role in advancing IR technologies, and the reusability of the created test collections has enabled the development of many effective retrieval algorithms that are used in many
modern search engine applications today.

Unfortunately, the Cranfield evaluation methodology, in its current form, can only be used for evaluating simple IR systems that return a ranked list of documents, and would encounter significant difficulty when applied to more sophisticated IR systems which have become increasingly popular due to the advancement in technologies for human-computer interaction. In particular, it is hard to use it to evaluate an interactive IR system where we need to assess the overall performance of a system over an entire interactive search session and compare two different search interfaces that may go beyond a ranked list of documents (e.g., an interface with features such as query suggestion or faceted browsing). Such sophisticated IR systems have so far been evaluated primarily through controlled user studies [40] or a proxy of such a user study experiment by performing search log analysis [65]. However, the experiment results obtained in such a way would be hard to reproduce due to the difficulty in completely controlling the users.

We propose a general formal framework for evaluating IR systems based on search session simulation that can be used to evaluate any IR system with reproducible experiments, including systems with sophisticated retrieval interfaces [15]. The key idea is to build “user simulators,” which are software programs that can simulate how a user would interact with a search engine (interface) when trying to finish a task. With a set of such user-task simulators, we can then test each IR system by having the system interact with the simulators. The interaction sequence of system responses and user actions can then be used to compute various quantitative measures of the system based on how effective the system has helped the (simulated) user finish a task.

We show that such a simulation-based evaluation framework is, in fact, a generalization of the traditional Cranfield evaluation method to enable reproducible experiments to evaluate or compare sophisticated IR systems. The current ranked list evaluation method can be derived quite naturally as a specific instantiation of the framework, where the simulated search session is a user sequentially browsing the presented search results.

One immediate benefit of the proposed framework is that it enables us
to examine any existing evaluation metric formally from the perspective of user simulation, which further helps reveal the exact assumptions a metric has made (often implicitly) about the simulated users. The analysis also helps provide an interpretation of any metric from a user’s perspective. We formally study several widely used measures, Precision, Recall, and Average Precision (AP), and reveal the assumptions made by these measures.

A more important benefit of the framework is that it would enable us to evaluate more complicated IR systems that are hard to evaluate with existing evaluation methods. As a case study to pursue this benefit, we build search session simulators to evaluate a set of tag-based search interfaces, a generalization of faceted browsing interfaces, with validation of our proposed framework from real user experiments and interesting findings about effectiveness of the interfaces for different types of users.

6.1 Search Simulation Framework

In this section, we formally characterize our proposed search simulation framework for interactive IR evaluation. We first explicitly define the basic components in the framework at the level of the whole interaction.

**Definition 6.1 (System, User, Task and Interaction Sequence).** In any interaction involving two parties issuing actions to each other in turn, we define the (interactive) system $S$ to be the party to be evaluated, the user $U$ to be the other party, the task $T$ to be the user’s information need, and the interaction sequence $I$ to be the whole process of the interaction.

A user may have different information need, or task, when using a system, and the user with a specific task may result in different interaction sequences due to the randomness of the user actions and the system responses.

**Definition 6.2 (Simulator).** A simulator is a (synthetic) user with a task, created for the purpose of evaluating a system.
In general, a system’s performance over an interaction sequence can be measured in two dimensions from a user’s perspective: reward and cost.

**Definition 6.3 (Interaction / Simulator Reward and Cost).** For an interaction sequence $I$ between a user $U$ with task $T$ and an interactive system $S$, the *interaction reward* $R(I,T,U,S)$ and the *interaction cost* $C(I,T,U,S)$ respectively represent the overall amount of reward and cost the user gets from the whole interaction.

For a simulator simulating a user $U$ with task $T$ and an interactive system $S$, the *simulator reward* $R(T,U,S)$ and the *simulator cost* $C(T,U,S)$ respectively represent the expected interaction reward and cost over all possible interaction sequences: $R(T,U,S) = E(R(I,T,U,S))$ and $C(T,U,S) = E(C(I,T,U,S))$, where the expectation is taken with respect to the distribution of all possible interaction sequences, $p(I|T,U,S)$.

Note that $p(I|T,U,S)$ would be entirely concentrated on a single interaction sequence if the interaction is deterministic.

The simulator reward $R(T,U,S)$ and cost $C(T,U,S)$ provide a complete and interpretable characterization of the utility of system $S$ to user $U$ with task $T$: $C(T,U,S)$ measures the effort made by a user, while $R(T,U,S)$ gives the reward that a user would receive for the effort. We chose to maintain reward and cost as two separate measures because the desired trade-off between them is inevitably application-specific, thus it should be treated as an external application of our framework. Moreover, we can easily further define the average utility and cost of a system over a group of simulators to obtain an overall reward and cost, or first combine reward and cost for each individual simulator and then compute the average over a group of simulators. These again would be better treated as applications of the framework. We will see some interesting examples in Section 6.2.

The formalism established above serves as a high-level framework for assessing interactive retrieval systems in general on the whole interaction level, in particular by evaluating the reward and cost of a task oriented user when interacting with the system through an interaction sequence. To assess the reward and cost at a finer level, we must define the interaction sequence in
more detail. To this end, we follow the Interface Card Model elaborated in Chapter 3 and partition the interaction between a user and an interactive IR system into a series of interaction laps:

**Definition 6.4 (Lap, Action and Interface Card).** The lap \( t = 1, 2, \ldots \) is the time unit of the interaction between a user and a system in which the user and the system each acts once in turn. In each lap \( t \), the user first issues an action \( a^t \), and the system then reacts by generating an interface card \( q^t \). The stopping action \( a'_{\text{stop}} \) is a special action the user could issue in each lap which ends the interaction.

It is often the case that there is certain level of intrinsic randomness in the user action and the system’s interface cards. In this work, we focus more on the user side, and we will later adopt a user action model describing the probabilistic distribution of the user actions at each lap.

When different users interact with the same system, or even when the same user interacts with the same system at different times, the user might tend to issue different actions, depending on e.g. the user’s habits, information need (task), and any past interactions between the user and the system. We characterize such user side information by user state (which we adopt from our sequential decision formulation of the ICM described in Chapter 4):

**Definition 6.5 (User State).** At each lap \( t \), the user state \( z^t \) denotes the collection of all the information that as a whole is sufficient to determine how likely the user issues each possible action given any interface card the system issues. The user state starts from the initial user state \( z^1 \), which depends on the user \( U \) and the task \( T \) and follows an initial user state distribution \( p_X(z^1) \). The user state then transitions across laps probabilistically via the user state transition function \( p_T(z^{t+1}|z^t, a^t, q^t) \).

Intuitively, the user state in many cases could be in the form of a multi-dimensional vector where each element denotes some aspect of the status of the interaction process, e.g. the stage of the interaction process, the remaining information need, etc. Based on the user state, we formalize the action model of the user:
**Definition 6.6** (User Action Model). The *user action model* specifies the probability distribution of the user issuing each possible user action in a given lap, where the probabilities are conditioned on the user state and the interface card: \( p(a^{t+1} | z^t, q^t) \).

We can now define the interaction on a finer level:

**Definition 6.7** (Interface Card Interaction Sequence). For an interaction process between a system \( S \) and a user \( U \) with task \( T \), the interaction sequence \( I \) is composed of the sequence of user states, the user actions and the interface cards in the whole interaction: \( I = ((z^1, a^1, q^1), (z^2, a^2, q^2), \ldots, (z^n, a^n, q^n)) \), where \( n \) denotes the total number of laps in the interaction. We define \( I^t \) to be the partial interaction sequence from lap 1 to lap \( t \), \( 1 \leq t \leq n \). (\( I = I^n \).)

Reward and cost can now be refined as follows.

**Definition 6.8** (Cumulative / Lap Reward and Cost). For user \( U \) with task \( T \), system \( S \) and interaction sequence \( I \), the *cumulative reward* and *cumulative cost* at lap \( t \) are respectively the total reward and cost the user obtains by the end of lap \( t \): \( R^t(I, T, U, S) = R(I^t, T, U, S) \), and \( C^t(I, T, U, S) = C(I^t, T, U, S) \). We define \( R^0(I, T, U, S) = C^0(I, T, U, S) = 0 \).

The *lap reward* and *lap cost* are respectively the difference of cumulative reward and cost between consecutive laps: \( r^t(I, T, U, S) = R^t(I, T, U, S) - R^{t-1}(I, T, U, S) \), and \( c^t(I, T, U, S) = C^t(I, T, U, S) - C^{t-1}(I, T, U, S) \).

The notion of cumulative reward and cost provides the basis for the simulator to track the reward and cost measures progressively along the interaction process. The lap reward and cost may depend on many factors related to the user’s current status and past interactions. To simplify the discussion, we assume that the user state contains the information sufficient to determine the lap reward and cost (in addition to the user action model) given any interface card:
Assumption 6.1 (Action Reward and Cost). The lap reward and lap cost are determined by the user’s action in the context of the user state and the system’s previous interface card, if any, and are also called the action reward and action cost: \( r^t(I, T, U, S) = r(a^t|z^{t-1}, q^{t-1}) \), \( c^t(I, T, U, S) = c(a^t|z^{t-1}, q^{t-1}) \).

We expand out the cumulative interaction reward and cost as a summation over action reward and cost, forming the computational basis for our proposed search simulation evaluation framework:

\[
R^t(I, T, U, S) = \sum_{i=1}^{t} r(a^i|z^{i-1}, q^{i-1}) \tag{6.1}
\]

\[
C^t(I, T, U, S) = \sum_{i=1}^{t} c(a^i|z^{i-1}, q^{i-1}) \tag{6.2}
\]

6.2 Analysis of Existing Metrics

In this section, we formally analyze some commonly used existing evaluation metrics using the proposed framework to reveal the (implicit) assumptions made underlying each measure and understand how we should interpret them based on the reward and cost defined on the user simulation.

We first instantiate the framework to obtain a general simulator for classical IR metrics:

Definition 6.9 (Classical IR simulator). The simulator’s task is to find relevant documents by going through a ranked list of documents. At each lap \( t \), the interface card is the document ranked at position \( t \). The user is assumed to sequentially browse the list and choose from three actions: click, skip or stop at each lap \( t \). We assume the simulator will always click a relevant document, and when seeing a non-relevant document, the user may skip or stop depending on the specific setting. The lap reward is 1 for a relevant document and 0 otherwise, and the cumulative reward is thus the number of relevant documents the simulator scanned through. The lap
cost is 1 for each document scanned by the simulator, and the cumulative cost is the total number of documents the simulator scanned through. The cumulative reward and cost are recorded in the user state.

The classical IR simulator serves as a common basis for further instantiations into specific simulators corresponding to each classical IR evaluation metric. In the following sections, we assume we have a test collection consisting of a number of queries and the relevance judgment labels of a set of documents with respect to each query, and our goal is to evaluate a ranked list of results generated by a system in response to a query. We will show that Precision, Recall, and Average Precision can all be interpreted in the context of our proposed framework when specific simulators are used. These simulators can help reveal the assumptions made by these measures and also provide interpretations of them from a user’s perspective.

We first examine precision and recall, two of the most fundamental metrics in IR.

**Definition 6.10 (Precision).** Given a list of retrieval results, the traditional measure Precision can be defined as the ratio of interaction award and cost, i.e., $R(I,T,U,S)/C(I,T,U,S)$, of a classical IR simulator that would never stop until having scanned through the whole result list.

The Precision Simulator shows clearly that Precision is focused on measuring the reward per unit of cost, but does not take into consideration of task completion; the task is not well specified, but the implied task can be assumed to be to find as many relevant documents as possible.

**Definition 6.11 (Recall).** Suppose there are $N$ relevant documents in the collection. Given a list of retrieval results, the traditional measure Recall can be defined as the task completion percentage $R(I,T,U,S)/N$, i.e. the interaction reward relative to the best possible interaction reward for perfectly completed task, for a classical IR simulator that never stops until having scanned through the whole list.
It is easy to see that the assumed task in the Recall Simulator is to find all relevant documents, though Recall is only focused on the collected reward, but does not measure the cost at all. Even if we combine Precision and Recall, there is still no direct measure of the cost, and the cost is only indirectly reflected in the Precision (relative to the reward). Interestingly, we can interpret the reciprocal of Precision as the average cost per relevant document (more generally, cost/reward ratio).

**Definition 6.12** (Precision@K / Recall@K). Precision@K and Recall@K are defined similarly as how Precision and Recall are defined except that such a simulator would stop when the accumulated cost (which is equal to the number of documents examined by the simulator) reaches $K$.

This definition shows that Precision@K and Recall@K can be interpreted as Precision and Recall with a “cost budget,” i.e., the simulated user wants to control the amount of effort. We can thus easily generalize both measures by allowing variable cost in examining each document/snippet (e.g., examining a longer document/snippet would have a higher cost) and using a cost threshold $\tau_c$, leading to Precision@$\tau_c$ and Recall@$\tau_c$, respectively.

We now examine one of the most important measures, Average Precision (AP). We first define the variable-recall simulator.

**Definition 6.13** (Variable-Recall Simulator). A variable-recall simulator is a classical IR simulator whose task is to collect $N'$ relevant documents, where $1 \leq N' \leq N$ ($N$ is the total number of relevant documents). The simulated user never stops scanning through the list until either the task is completed or the list is exhausted.

**Definition 6.14** (Average Precision). In the simulation framework, Average Precision can be defined as the average ratio of the interaction reward and cost: $R(I, T, U, S)/C(I, T, U, S)$ for a set of $N$ variable-recall simulators, each with the task of collecting $1, 2, \ldots, N$ relevant documents, respectively.

By examining AP in the simulation framework, we see that AP should be interpreted as the average performance of a system on a set of different
retrieval tasks or different simulated users. While the Precision and Recall simulators only simulate a single user/task, the AP simulator simulates a set of users with variable recall demand; this explains why AP is more discriminative than Precision/Recall, and is thus also more suitable for comparing two ranked lists. This analysis result further suggests that in general, we can systematically vary the parameter of any simulator (recall in the case of AP) to obtain more discriminative measures that can better detect even the smallest differences between two ranking methods; AP is only one of the many such possibilities and may not necessarily be the best one.

The variant of AP@K documents can be easily derived by setting a cost budget for all the simulated users as in the case of Precision/Recall@K.

Many other evaluation metrics such as Mean Reciprocal Rank (MRR) [45], Ranked-Based Precision (RBP) [38], Normalized Discounted Cumulative Gain (NDCG) [66], time-based measure [46], can also be studied rigorously in the proposed framework to reveal their assumptions about users and tasks. For example, MRR is obtained when a precision simulator has a task of only finding one relevant document (and then stop). RBP assumes, on top of the precision simulator, a constant stopping rate at each position of the ranked list. In NDCG, the discounting factors for each ranked position also correspond to the simulator’s stopping rate at each position, and the overall gain calculated is the simulator’s expected reward over all stopping positions. The time-based measure is closely related, only except that the probability of stopping depends on the time spent into the search session (i.e., time cost) instead of on the lap count.

We could also easily extend our instantiations to generalizations of evaluation metrics on session search. For example, Session NDCG [67] could be derived similarly as classic NDCG, only with the additional simulator action model for continuing / abandoning the search after scanning through the document list of each query in the session. The U-measure based on trail-text introduced in [68], as another example, could be derived from our proposed framework by dividing the simulator’s interaction with the system into word-level laps, and the simulator may abandon the search after reading till each word (e.g. in snippet, document, etc.).
The great generality of our framework is not a coincidence; it is a natural consequence of the basis of our framework - the simulator and the reward/cost measures - which are the minimal basis that maps to real world users and what they care about in an IR system; all existing metrics tried to achieve the same goal but with additional simplification assumptions for the sake of computational convenience. In particular, for example, our analysis based on the simulator models suggest that one major class of assumption underlying the existing evaluation metrics is on when and how likely the user stops throughout the interaction, and every assumption has its own advantages as well as drawbacks when compared with real user behaviors. A very important future direction is thus to study users’ stopping tendencies more rigorously and propose more realistic user stopping action models, which can then be used in the proposed framework to derive potentially more meaningful metrics than the existing ones.

6.3 Simulated Evaluation on Tag-based Search Interface

In this section, we apply our proposed evaluation framework on interactive retrieval systems that do not follow a simple ranking interface, and show that an instantiation of our general framework could lead to novel evaluation method for interactive systems where no traditional evaluation methodology could be applied in a principled way.

We focus on a set of interactive retrieval interfaces where, in addition to lists of documents, tags related to the document contents are used to facilitate user navigation. A common example of such tag-based search interfaces is the faceted browsing interface, where facet filters serve as tags to help users zoom into specific subsets of the documents. The Interface Card Model proposed in Chapter 3 led to a novel method for optimizing tag-based search interfaces via automatically adjusting the interface layout based on the screen size and the estimated user interest. To evaluate and compare these relatively more sophisticated interactive retrieval interfaces, traditional evaluation methodologies focusing mainly on assessing ranked lists of documents could not be
easily applied, because the user-system interactions do not adopt a sequential scanning manner. This is also the reason why we could only rely on real user experiments for the comparison experiments in Chapter 3. In this section, as an example of demonstrating the effectiveness of our proposed search simulation framework, we show that an instantiation of the proposed framework could lead to reasonable evaluation practices of the search interfaces on different types of users (or simulators), and we also validate the simulation by comparing the simulator behaviors with real user behaviors.

To instantiate the search simulation framework into a simulator model for the tag-based search interfaces, we assume that each screen the simulator sees is an interface card; the simulator could either select a document or a tag (if shown) on the screen, or click some other control buttons (e.g. scroll down / next page) to look for new content, and then the system displays a new interface card to the user and the interaction goes on. In the traditional faceted-browsing interfaces, the interface layout is static: on a moderate sized screen, there is typically a tag list on the left and a document list on the right, where the user could either scan through the documents or scan through the tags to narrow down the set of documents shown on the right; on a very small screen (e.g. of a smart phone), only one of the two lists (i.e. the tag list or the document list) could be displayed at a time, and there usually is an extra button for the user to switch between the two lists. On the contrary, the interfaces proposed in the Interface Card Model, which we designate by “ICM interfaces”, automatically adjust their layouts (e.g. between only showing tags, only showing documents, showing half-screen tags and half-screen documents, etc.), and the user either clicks a shown document / tag or click “next page” in each interaction lap.

We define the instantiation of our proposed search simulation framework for the case of tag-based search interfaces as follows:

**Definition 6.15 (Tag-based search interface simulator).** A tag-based search interface simulator $U$ is assumed to be interested in one or a few documents in the collection, which are designated by the simulator’s target document(s). The simulator’s task $T$ is to find all target document(s). The simulator’s action model on the interface cards in a tag-based search interface is defined as follows (assuming $\tau$, $\tau_1$ and $\tau_2$ are constants between 0 and 1):
• If the simulator sees a target document, they always click it, and in cases of multiple target documents, they click one of them uniformly randomly.

• Otherwise, if the simulator sees a tag related to a target document, they always click it, and in cases of multiple related tags, they click one of them uniformly randomly.

• Otherwise, they seek for the next card in a way depending on the type of the interface:
  
  – On an ICM interface, they always click next card;
  
  – On a moderately sized traditional static interface displaying both a tag list and a document list, the simulator scrolls down the document list with probability $\tau$ (designated as the document tendency value) and scrolls down the tag list with probability $(1 - \tau)$;
  
  – On a very small traditional static interface displaying only a tag list or a document list, if the simulator faces a document list (which is usually the case for the initial interface card), they scroll it down with probability $\tau_1$ (designated as the document inertia value) and switch to the tag list with probability $(1 - \tau_1)$; if the simulator faces a tag list, they scroll it down with probability $\tau_2$ (designated as the tag inertia value) and switch to the document list with probability $(1 - \tau_2)$.

• The simulator only and always stops when all target documents are found.

The lap cost is 1 for each lap the simulator undergoes, and the overall evaluation metrics is the simulator’s interaction cost $C(I, T, U, S)$ for completing the task.

The implicit user state of the simulator is the task, i.e. the set of target documents, plus, for interacting with the very small static interface, the additional binary status of whether the user is browsing the document list or the tag list. The parameters $\tau$, $\tau_1$ and $\tau_2$ could be very different for different types of users, and could be learned from user search logs.
Such an instantiation is apparently an overly simplified model for users in the real world, and it could be easily extended in a lot of aspects to reflect more realistic settings (e.g. with consideration of information scent when the simulator decides on what link to follow). As the very first example of instantiating our proposed search simulation framework, we stick with this simplified simulator model and demonstrate that it could lead to fairly reasonable and interesting evaluation results, leaving further extensions of the simulator to future research work.

6.3.1 Simulated Evaluation

We implemented the tag-based search interface simulators and use them to evaluate and compare the static interfaces and the ICM interfaces on a medium screen as well as on a small screen, where we used the New York Times API [59] to obtain news articles and keywords respectively as our documents and tags. The medium screen could hold up to 2 documents or 8 tags; on the static interface, 1 document alongside 4 tags are displayed at a time. The small screen could hold up to 1 document or 4 tags; on the static interface, the (simulated) user needs to switch between the document list and the tag list. We vary the number of documents in the collection (obtained through the API calls) as well as the parameters $\tau$, $\tau_1$ and $\tau_2$. We assume the simulator is interested in only one (uniformly randomly selected) document in the collection in each search session, and we record down the average number of laps for the simulator to find the target document across multiple simulated sessions, which is an unbiased estimate of the simulator’s interaction cost.

Medium screen

Figure 6.1 shows the interaction cost against different document tendency values $\tau$ on a medium screen with the static interface, and we set the number of documents in the collection to be either 30 or 100 in the experiments. It is firstly not surprising to find that the interaction cost on a collection of 30
items is always lower than on a collection of 100 items across all \( \tau \) values, as it naturally takes less laps for the simulator to navigate in a smaller collection. It could also be observed that the cost tends to grow higher when \( \tau \) is either too small or too large, suggesting that it is not a good idea for the simulator to stick too much to the document list, i.e. \( \tau \) is too large, or too much to the tag list, i.e. \( \tau \) is too small. Such an implication makes sense: sticking too much to the document list is essentially giving up the “zoom-in” functionality provided by the tags, whereas sticking too much to the tag list makes the simulator pay too little attention to the documents, which are after all what the simulator is really looking for. It is also interesting to observe that the negative effect from sticking to the document list (i.e. \( \tau \) is large) is weaker when the collection size is small, which is reasonable as keep scrolling through a small collection is not a problem as serious as keep scrolling through a large collection.

Note that the curves are observed to fluctuate a lot around their overall trends, since the effectiveness of the tags (news keywords) in helping the simulator narrow down to specific documents (news articles) could vary significantly depending on the specificity of the tags. Such fluctuations will also be seen in the other experiments we report.
To use our simulators to compare the static interface with the ICM interface, we set $\tau = 0.3$ for the static interface, and Figure 6.2 shows the simulation result on both interfaces with various number of documents in the collection. Despite the expected fluctuations, we clearly observe that the ICM interface achieves more efficient navigation across all #documents than the static interface, and the interaction cost grows at a slower pace in the ICM interface than in the static interface as the collection size grows. We also tried setting $\tau$ to other nearby values and obtained similar results. Such comparison outcomes coincide with the findings in Chapter 3.

Small screen

On a small screen with static interface, there are two parameters, the document inertia $\tau_1$ and the tag inertia $\tau_2$, underlying the simulator’s action model. Figure 6.3 shows the interaction cost for different combinations of $\tau_1$ and $\tau_2$ on top of a collection of 30 and 100 documents, with brighter color for lower cost and darker color for higher cost. In addition to what we observed on the medium screen - the cost in navigating through a smaller collection is lower than that in navigating in a larger collection - there are a couple of interesting findings unique to the small screen. Firstly, for both collec-
tion sizes, the cost is generally lower when the tag inertia is relatively high ($\tau_2 \geq 0.7$), i.e. when the simulator tends to scan more tags before switching back to the document list. It is a reasonable strategy for the simulator to keep scanning through more tags, since discovering a good tag would eventually shrink the number of documents to look through even though it takes a few more scrolls on the tag list in the short run. Secondly, given a relatively high tag inertia $\tau_2$, it is a good idea to keep the document inertia low in the smaller collection ($\tau_1 \leq 0.6$), while it is better to raise it higher in the larger collection ($\tau_1 \geq 0.5$). Such a finding also makes intuitive sense: when the document collection grows larger, the simulator should be more patient in scrolling through the document list rather than constantly jumping back to the tag list.

To compare the static interface with the ICM interface on the small screen, we set $\tau_1 = 0.5$ and $\tau_2 = 0.8$ for the simulator, and Figure 6.4 shows the interaction cost for the simulator on the two interfaces across different collection sizes. The comparison result is analogous to the one for the medium screen: the ICM interface achieves lower cost than the static interface, and the cost also grows slower on the ICM interface as the collection grows. The finding again coincides with those found in Chapter 3.
6.3.2 Validation from Real User Experiment

We conducted real user experiments on Amazon Mechanical Turk [60] following the scheme described in Chapter 3, and compare real user behaviors with the behaviors of our simulators. We gave users the task of finding a target news article and asked them to navigate through the static interface and the ICM interface, on both medium screens and small screens, and we record down the users’ clicks throughout the interaction. On the medium sized static interface, we compute the users’ average rate of choosing to scroll the document list across all laps as $\hat{\tau}$; on the small static interface, we compute the users’ average rate of continuing scrolling through the list across all document screens and all tag screens as $\hat{\tau}_1$ and $\hat{\tau}_2$, respectively. Table 6.1 displays the result.

<table>
<thead>
<tr>
<th>Screen size</th>
<th>Sample size</th>
<th>Workers’ average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>38</td>
<td>$\hat{\tau} = 0.211$</td>
</tr>
<tr>
<td>Small</td>
<td>42</td>
<td>$\hat{\tau}_1 = 0.845$, $\hat{\tau}_2 = 0.370$</td>
</tr>
</tbody>
</table>

Table 6.1: Real user action averages

It could be observed that on the medium static screen, the users have a relatively low tendency ($\hat{\tau} = 0.211$) on average to stick to scrolling the
document list, and such a $\tau$ value also led to a fairly good interaction cost measure as observed in Figure 6.1. In other words, the real users are generally able to utilize the tags nearly optimally in facilitating their navigation on the medium static screen. On the small static screen, on the other hand, the users have a high inertia ($\tau_1 = 0.845$) of keeping scrolling through the document list, but a relatively low inertia ($\tau_2 = 0.370$) of scrolling through the tag list. Such a combination of $\tau_1$ and $\tau_2$ values resides in the lower-left portion in the two heat maps in Figure 6.3, which results in sub-optimal interaction cost measures. The users navigating on the small static interface do not tend to switch to the tag list when they are scrolling through the documents, and even when they switch to the tag list, they quickly switch back to the document list without exploring more tags when they could not find a relevant tag. The reason is most likely that the small screen only has space for either the document list or the tag list, and is initially showing the document list, so a lot of users merely follow the document list, and might only consider the switch as a glimpse of what tags might be there and do not recognize the power of exploring more tags; on the contrary, the medium screen always displays both the documents and the tags, so the users are free to explore both lists without taking any extra effort in switching between the lists.

In Chapter 3, real user experiments were conducted to compare the ICM and static interfaces on small and medium screens, and it was concluded that the ICM interface is more efficient in helping users navigate, and also that the benefit of ICM over the static interface is more striking on the small screen than on the medium screen. In our experiment and analysis, with the tag-based search interface simulator as an extension of our proposed search simulation framework, we reached the same conclusion that the ICM interface is better as shown in Figure 6.2 and 6.4, which validates that our proposed search simulation framework could reliably assess the effectiveness of search interfaces. More interestingly, by comparing the real users’ actions with the spectrum of our simulators’ action model, we observe that the real users adopt a nearly optimal strategy on the medium screen yet a sub-optimal strategy on the small screen, which are novel insights into the reason why the difference between ICM and static interfaces in user navigation efficiency is more significant on the small screen as concluded in Chapter 3. Such novel
insights would be hardly possible to draw without establishing the proposed search simulation framework. These results also highlight another important benefit of the proposed simulation framework for understanding user behavior in detail by fitting simulators to real user interaction log data.

6.4 Summary

We presented a new general framework for evaluating arbitrary information retrieval (IR) systems based on search session simulation. The motivation for this framework is to enable reproducible experimental evaluation of sophisticated IR systems, particularly interactive IR systems, in the same spirit as the Cranfield evaluation methodology. The main idea is to generalize the current Cranfield evaluation method based on a test collection to one based on a set of user-task simulators and measures defined on a whole interaction session. We examine multiple commonly used measures in IR evaluation in this framework and show that they can all be derived as special cases of the framework under various assumptions about the user that they (implicitly) intend to simulate. Analysis of these assumptions reveals insights about how to improve these measures, which not only are practically useful, but also point out interesting new research directions. We also propose a way to construct user simulators for evaluating a set of tag-based search interfaces, and conduct simulation experiments to assess the effectiveness of different interface layout strategies. We show that such systems, which cannot be evaluated using any existing method in a principled way, can now be evaluated using the constructed simulators with interesting observations.

The proposed framework lays a theoretical foundation for experimental studies of sophisticated IR systems and opens up many new research directions. For example, we can use the framework to derive potentially better metrics than the existing ones that we analyzed, and to further analyze many more evaluation metrics of various tasks. The framework also opens up many interesting opportunities to leverage search log data to build various realistic user simulators for evaluating potentially very complicated search systems.
CHAPTER 7

CONCLUSION

We proposed a general formal framework for optimizing and evaluating interactive information retrieval systems. In the Interface Card Model (ICM), we view the interactive retrieval process as a process of a system playing a cooperative card game with a user with the goal of minimizing the user’s effort and maximizing the user’s gain of relevant information. Further, we propose a novel refinement of the Interface Card Model based on sequential decision theory, i.e. ICM-US, that can facilitate formal user modeling and naturally connect optimization of interactive retrieval with Markov Decision Process and Reinforcement Learning (RL) in a general way, thus enabling the use of RL to solve potentially a wide range of problems of optimal interface design. As a specific example to apply our proposed interface optimization framework in a larger scale real world application, we propose a Bayesian framework for user preference modeling and dynamically optimizing a faceted browsing system based on users’ facet selection interactions. Finally, we generalize the current Cranfield evaluation method and present a new general framework for evaluating arbitrary information retrieval systems based on search session simulation, enabling reproducible experimental evaluation of sophisticated IR systems, particularly interactive IR systems.

This thesis introduces a novel and very broad foundation for multiple research domains to collaborate on automatic optimization of interactive systems and interfaces. First, research efforts in information retrieval can bring better information relevance assessment and content understanding to the proposed framework and enable better reward estimation. Second, machine learning research can help refine the reinforcement learning methodologies for adaptive interface optimization, and can also improve user modeling research for better reward, cost and action model estimations. Third, the constraint for the interface cards and the set of user actions in the proposed framework
is directly related to human-computer interaction research, where improved understanding of human perception and novel interface technologies can potentially introduce very different and interesting means of user interactions with the system as well as constraints of the interface cards. Finally, research in economics on user decision making and cost-benefit analysis can help improve the user action model estimation. This thesis marks a starting line instead of a finishing line, and it is a very promising and exciting road ahead.
REFERENCES


